



Trabajo de Fin de Grado  
Adrian Valdivieso Paredes.

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)



## Contenido

1.	Resumen .....	6
2.	Introducción .....	7
2.1.	Contexto .....	7
2.2.	Motivación .....	7
2.3.	Objetivos.....	7
2.4.	Tecnologías utilizadas.....	8
2.4.1.	<i>Angular</i> para el Front-end.....	8
2.4.2.	<i>Node.js</i> para el Back-end.....	8
2.4.3.	<i>Sequelize</i> para la interacción con la base de datos.....	8
2.4.4.	<i>MySQL</i> con <i>Workbench</i> para la base de datos.....	9
2.4.5.	<i>Draw.io</i> para la generación de los diagramas UML.....	9
2.4.6.	<i>GitHub</i> para control de versiones.....	9
2.4.7.	<i>Visual Studio Code</i> como entorno de desarrollo principal.....	10
2.4.8.	<i>PostMan</i> como herramienta de pruebas hacia las rutas creadas.....	10
2.4.9.	<i>Amazon web service</i> se utilizará para alojar la base de datos.....	10
2.4.10.	<i>Bootstrap</i> se utilizará para alojar la base de datos.....	11
2.4.11.	<i>Instagantt</i> se utilizará para llevar un control del proceso de desarrollo.....	11
3.	Análisis del sistema.....	12
3.1.	Sistema inicial.....	12
3.2.	Requisitos funcionales.....	14
3.3.	Identificación de los actores.....	16
3.3.1.	Usuario nominal .....	16
3.3.2.	Usuario Monitor.....	17
3.3.3.	Usuario Administrador .....	18
4.	Arquitectura del sistema.....	19
4.1.	Diagrama Entidad Relación Previo. ....	19
4.2.	Diagrama Entidad Relación Actualizado. ....	19
4.3.	Diagrama Relacionable Previo.....	19
4.4.	Diccionario de datos. ....	20
4.5.	Explicación de las relaciones entre tablas.....	26
5.	Diseño e implementación del sistema.....	27

5.1.	Diseño del servidor .....	27
5.1.1.	Estructura de carpetas .....	27
5.1.2.	Archivo principal app.js.....	28
5.1.3.	Variables de entorno. ....	29
5.1.4.	Archivo de configuración con la base de datos.....	30
5.1.5.	Carpeta Models.....	30
5.1.6.	Carpeta Routes. ....	32
5.1.7.	Carpeta Middlewares. ....	34
5.1.8.	Carpeta Controllers. ....	35
5.1.9.	Carpeta Controllers. ....	38
5.1.10.	Carpeta Validadores. ....	42
5.1.11.	Estructura.....	42
5.2.	Diseño del cliente. ....	43
5.2.1.	Carpeta Components. ....	45
5.2.2.	Carpeta Guards.....	45
5.2.3.	Carpeta Interceptors.....	46
5.2.4.	Servicios. ....	46
5.2.5.	Carpeta Models.....	47
5.2.6.	Carpeta Pipes .....	47
5.2.7.	Archivo Environments .....	47
5.2.8.	Carpeta Utils .....	48
5.2.9.	Estructura.....	48
5.3.	Conexión con los servicios RDS de Amazon Web Service.....	48
5.3.1.	Pasos para implementar el servicio RDS de Amazon Web Service .....	49
6.	Pruebas de funcionamiento y resultados obtenidos. ....	53
6.1.	Tabla de verificación de requisitos funcionales.....	53
6.2.	Pruebas realizadas. ....	54
7.	Conclusiones. ....	54
7.1.	Resultados obtenidos. ....	54
7.2.	Ampliaciones y mejoras. ....	54
7.2.1.	Enviar Errores a Slack.....	54
7.2.2.	Servicio EC2 de Amazon Web Service.....	54
7.2.3.	Incorporación de Tailwind y ShadCN. ....	55

7.3.	Estimación del tiempo empleado.....	55
7.4.	Valoración personal.....	56
8.	Bibliografía.....	58
9.	Anexo 1 Diagramas.....	60
10.	Anexo 2 Pruebas de Funcionamiento.....	66
10.1.	La página tiene que tener un uso destinado para un usuario y un uso destinado para un profesional.....	66
10.2.	Registro de usuarios.....	66
10.3.	Inicio de sesión.....	67
10.4.	Para acceder a todas las funcionalidades de la página se necesitará un registro previo.	68
10.5.	Crear actividades para los usuarios.....	69
10.6.	Todo tiene que estar en modo de lectura fácil.....	69
10.7.	Notificar a los usuarios de las actividades.....	70
10.8.	Lista de espera.....	70
10.9.	Crear formularios para la valoración de las actividades.....	71
10.10.	Distintos programas.....	72
10.11.	Los profesionales podrán ver una lista de todos los usuarios registrados en la página web.	73
10.12.	Diseño corporativo.....	74
10.13.	Vista para el usuario con las actividades los siguientes siete días.....	75
10.14.	Google Calendar.....	76
11.	Anexo 3 Manual de Usuario.....	77
11.1.	Descripción del documento.....	77
11.2.	Inicio de sesión.....	77
11.3.	Página principal usuario Nominal.....	79
11.4.	Página principal usuario Monitor.....	81
11.5.	Página principal usuario Administrador.....	82
11.6.	Página de horarios Nominal.....	83
11.7.	Página de horarios Monitor y Administrador.....	85
11.8.	Pantalla Tu Horario.....	87
11.9.	Pantalla ver usuarios Nominal y Administrador.....	89
11.10.	Pantalla lista de actividades Administrador.....	90



11.11.	Pantalla lista de campañas Administrador.....	92
11.12.	Pantalla lista de tipos de actividades Administrador.....	94
12.	Anexo 4 Documentación de la API.....	96
13.	Anexo 5 Código fuente.....	96
14.	Índice de imágenes .....	97
15.	Índice de diagramas .....	99
16.	Índice de pantallas. ....	99
17.	Índice de pruebas de funcionamiento. ....	100



## 1. Resumen.

El Trabajo de Fin de Grado se basa en la aplicación del conocimiento adquirido a lo largo de estos dos cursos, donde se combina la teoría con la práctica para desarrollar una solución integral que aborde los requerimientos del cliente, en este caso Plena Inclusión.

Este proyecto implica un análisis de los requisitos del cliente, seguido por la planificación y ejecución de cada etapa del desarrollo de software, desde la creación del diseño del Front-End hasta la implementación del Back-End y la estructura de la base de datos, todo ello llevado de la mano junto con tecnologías para la gestión de versiones.

El documento que acompaña al proyecto proporcionará una descripción detallada de cada fase del proceso, incluyendo la justificación de las decisiones tomadas en cuanto a las tecnologías utilizadas, la identificación y resolución de desafíos técnicos.

## 2. Introducción.

### 2.1. Contexto.

La idea nace de la necesidad que presenta la asociación llamada “Plena Inclusión” (Plena Inclusión Aragón, 2008), de mejorar la página web donde tiene alojada toda su información. Ya que por el momento la página web se encuentra desactualizada y necesita varias mejoras, así como la agregación de nuevas funcionalidades dentro de la misma para que tenga un uso más adecuado por parte de los usuarios.

Estos usuarios presentan características especiales, sobre todo en la discapacidad intelectual de los mismos. Por esa misma razón, toda la parte del proyecto enfocada al usuario final (Front-End) tendrá un formato especial y de lectura fácil.

### 2.2. Motivación.

La motivación de llevar a cabo este proyecto se encuentra en el deseo de querer ayudar a una asociación sin ánimo de lucro, a mejorar su página web. Ya que, las asociaciones de este tipo ayudan a muchas personas que lo necesitan y no esperan nada a cambio, a su vez, realizando una excelente labor con sus programas, en este caso hacia las personas con discapacidad cognitiva.

### 2.3. Objetivos.

El objetivo del proyecto es desarrollar una plataforma web para una asociación que defiende los derechos de las personas con discapacidad intelectual en España. La página se centrará en informar sobre actividades y noticias relevantes, así como en facilitar el registro de usuarios y profesionales. Se establecerán tres tipos de usuarios con diferentes niveles de acceso y funcionalidades.

Además, se garantizará un proceso de registro asistido para los usuarios y se implementará un sistema de notificaciones para recordarles las próximas actividades. La plataforma se diseñará con un enfoque en la accesibilidad, utilizando el modo de lectura fácil y proporcionando imágenes descriptivas. También se permitirá a los usuarios evaluar las actividades y se organizarán en distintos programas a lo largo del año para una mejor organización.

El diseño de la página seguirá el estilo corporativo de la asociación y se integrará la funcionalidad de Google Calendar para una mejor gestión de las actividades.

## 2.4. Tecnologías utilizadas.

### 2.4.1. Angular para el Front-end.

Angular (Hevery, 2016) es un *framework* de desarrollo de aplicaciones web desarrollado y mantenido por Google. Se elegiría Angular debido a su estructura modular, su capacidad para crear aplicaciones de una sola página (SPA) y su robusto sistema de componentes, que facilita la construcción de interfaces de usuario dinámicas e interactivas. Además, Angular ofrece un sólido soporte para la creación de aplicaciones accesibles, lo que encajaría perfectamente con el requisito de lectura fácil de la asociación.



Imagen 1 Angular

### 2.4.2. Node.js para el Back-end.

Node.js (Lienhart Dahl, 2009) es un entorno de ejecución de JavaScript del lado del servidor que permite desarrollar aplicaciones web escalables y eficientes en términos de recursos. Se elegiría Node.js debido a su naturaleza basada en eventos y su capacidad para manejar múltiples solicitudes de forma asíncrona, lo que lo hace ideal para aplicaciones web en tiempo real y que requieren una alta concurrencia.



Imagen 2 NodeJS

### 2.4.3. Sequelize para la interacción con la base de datos.

Para la interacción con la base de datos MySQL, se utilizará Sequelize (Depold, 2012), un ORM (Mapeador de Relaciones de Objetos) de Node.js que simplifica la gestión de la base de datos mediante la representación de las tablas como modelos de datos y el uso de consultas de alto nivel. Sequelize agiliza el proceso de desarrollo al proporcionar una capa de abstracción sobre la base de datos y garantiza la seguridad y consistencia de los datos. Su uso en conjunto con Node.js optimiza la eficiencia y la productividad del desarrollo de aplicaciones web.



Imagen 3 Sequelize



#### 2.4.4. MySQL con Workbench para la base de datos.

MySQL (Widenius et al., 2001) es un sistema de gestión de bases de datos relacional de código abierto ampliamente utilizado en la industria. Se elegiría MySQL debido a su estabilidad, rendimiento y amplia adopción. Además, Workbench (Widenius et al., 2005) proporciona una interfaz gráfica intuitiva para diseñar, modelar, administrar y administrar bases de datos MySQL, lo que facilita el desarrollo y el mantenimiento de la base de datos para el proyecto.



*Imagen 4 MySQL*

#### 2.4.5. Draw.io para la generación de los diagramas UML.

Utilizamos draw.io (Subins2000, 2017) para visualizar y diseñar la estructura de la base de datos MySQL, representando las entidades como tablas y estableciendo las relaciones entre ellas de manera clara y comprensible. Esto nos permitiría tener una visión global de la arquitectura de la base de datos y asegurarnos de que esté correctamente diseñada para satisfacer las necesidades del proyecto.



*Imagen 5 Draw.io*

#### 2.4.6. GitHub para control de versiones.

GitHub (Tom-Preston-Werner et al., 2008) se utilizará como una plataforma centralizada para el control de versiones del código fuente. Además, se aprovecharán sus herramientas de gestión de proyectos para realizar seguimiento del progreso y así tener versiones de respaldo por si se quiere ver la evolución de proyecto.



*Imagen 6 GitHub*



#### 2.4.7. *Visual Studio Code* como entorno de desarrollo principal.

Visual Studio Code (Microsoft, 2015) sería la elección óptima para este proyecto debido a su potencia, versatilidad y facilidad de uso. Este entorno de desarrollo integrado (IDE) ofrece un sólido soporte para JavaScript, HTML y CSS, los cuales son fundamentales para el desarrollo tanto del frontend como del backend. Además, su integración nativa con Git simplifica el control de versiones y la colaboración en equipo.



Imagen 7 Visual Studio Code

#### 2.4.8. *PostMan* como herramienta de pruebas hacia las rutas creadas.

PostMan (Abhinav Asthana & Sobti, 2012) se utilizará para probar la API de una manera más sencilla permitiendo enviar peticiones y analizar las respuestas por parte de la misma, además se utilizará para generar la documentación de todas las rutas de acceso a la información.



Imagen 8 PostMan

#### 2.4.9. *Amazon web service* se utilizará para alojar la base de datos.



Imagen 9 Amazon Web Service

Amazon Web Service (Matt Garman & C.J. Moses, 2002) proporciona un servicio de almacenamiento de datos llamado *Amazon RDS*, se trata de un servicio que facilita la configuración y escalado de base de datos relacionales en la nube. Se ha decidido alojar la base de datos de este proyecto en este servicio proporcionado por *Amazon Web Service*.



2.4.10. *Bootstrap* se utilizará para alojar la base de datos.

Bootstrap (Mark Otto & Jacob Thornton, 2011) se utilizará para gestionar el diseño de la aplicación, debido a que ofrece herramientas para dotar a la aplicación de un aspecto profesional, manteniendo una buena legibilidad y adaptabilidad a varios tamaños de dispositivos.



*Imagen 10 Bootstrap*

2.4.11. *Instagantt* se utilizará para llevar un control del proceso de desarrollo.

*Instagantt* (Dustin Moskovitz & Justin Rosenstein, 2008) Es una herramienta de gestión de proyectos en línea para ayudar a los equipos a planificar, visualizar y gestionar los proyectos mediante diagramas de Gantt. Los diagramas de Gantt son gráficos de barras que representan un cronograma de proyectos, mostrando tareas, fechas de inicio y finalización y el progreso del proyecto.



*Imagen 11 Instagantt I*

### 3. Análisis del sistema.

#### 3.1. Sistema inicial.

Se parte de un trabajo hecho previamente, se tiene una primera instancia del proyecto el cual deja varios aspectos de la aplicación por resolver.

Esta primera versión está realizada con React (Jordan Walke, 2013), que es una de las librerías más usadas y populares de JavaScript para el desarrollo de aplicaciones web y móviles, junto NodeJs para la gestión del Back-end y conexión con base de datos con MySQL.

Dentro de esta versión se han detectado varios problemas, tanto en el cliente, como en el servidor:

- No existe una gestión que asegure el acceso hacia las rutas del servidor.
- No existen validaciones de datos provenientes de solicitudes dentro del servidor.
- Falta de gestión de archivos multimedia.
- El servidor no tiene la estructura de una API restful.
- Por parte del cliente no se definen componentes estructurados y compartidos en la aplicación entera.
- No se tratan los datos que se envían al servidor de una manera eficaz.
- En general no se hace buen uso de las herramientas que React proporciona.
- A la base de datos le falta una gestión más adecuada para satisfacer las necesidades del cliente.

Tras entender que la primera versión de la aplicación tiene varios aspectos por resolver, se ha llegado a la conclusión de rehacer completamente la aplicación. Todo esto para garantizar el buen funcionamiento de la misma.

Por parte del Front-End, se va a cambiar de tecnología a usar. En este caso se hará uso de Angular, que como anteriormente se ha señalado, es un framework robusto y bien estructurado para desarrollar aplicaciones SPA (*Single Page Application*), lo que permite mejorar la experiencia del usuario al proporcionar una interfaz rápida y reactiva.

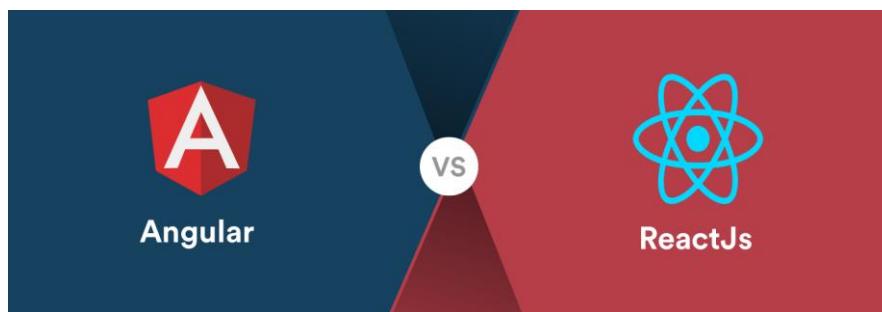


Imagen 12 Angular y React

Todo esto ofreciendo una estructura modular, herramientas integradas para el uso de las rutas, pruebas y gestión de dependencias, de esta manera facilitando el desarrollo, escalabilidad y mantenimiento de aplicaciones complejas.

Por parte del Back-End, se decide mantener NodeJs, añadiendo las funcionalidades que se han visto escasas. Se ha decidido seguir las reglas API REST para construir la parte del servidor, esto quiere decir que las rutas por parte del servidor van a seguir un estándar basado en los principios fundamentales de REST.

Estos incluyen la utilización de recursos, identificados por URLs únicas y la manipulación de estos recursos a través de métodos HTTP estándar, como pueden ser los métodos: GET, POST, PUT y DELETE, todo esto acompañado de documentación sobre los mismos recursos generada por *PostMan*.

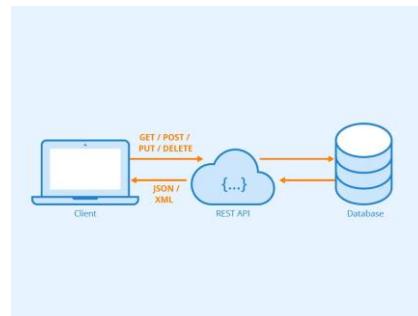


Imagen 13 API REST

Con respecto a la gestión de base de datos, se ha analizado con detenimiento el funcionamiento de la misma y se han corregido, editado, borrado y añadido tablas y campos necesarios para llevar a cabo la ejecución del proyecto.

Para ello se ha elaborado nuevamente un diagrama entidad relación y un diagrama relacionable que satisfagan por completo las necesidades del proyecto. También se ha decidido utilizar una base de datos alojada en la nube, para facilitar el acceso a los datos desde cualquier dispositivo donde se ejecute la aplicación.

### 3.2. Requisitos funcionales.

El cliente nos ha pedido una serie de funcionalidades las cuales se tiene que ver reflejadas en el desarrollo del proyecto. A continuación, se va a detallar cada una de las funcionalidades:

- **La página tiene que tener un uso destinado para un usuario y un uso destinado para un profesional**

Esto es debido a que un usuario solo podrá utilizar la página a modo informativo, pudiéndose apuntar a las actividades disponibles o desapuntarse de las mismas o echando un vistazo a las nuevas noticias en el apartado de las noticias. En cambio, un profesional podrá añadir, editar, eliminar actividades, así como registrar a nuevos usuarios.

- **Para el registro de las personas**

Los profesionales se registrarán de manera manual, dando todos los datos necesarios para su uso, en cambio los usuarios, debido a su condición, necesitarán de un profesional a la hora de su registro por lo que será el profesional quien se encargue de registrar a un nuevo usuario y de entregarle la contraseña con la que acceder a la página.

- **Inicio de Sesión**

El inicio de sesión se hará mediante el correo electrónico del usuario y su contraseña, pero en la última actualización se ha decidido hacer el inicio de sesión mediante la autenticación de Google.

- **Para acceder a todas las funcionalidades de la página se necesitará un registro previo**

Con esto nos aseguramos que las personas que se apuntan a las actividades disponibles son personas interesadas en ellas y no cualquier persona.

- **Crear actividades para los usuarios**

Los profesionales dispondrán de una funcionalidad para crear actividades donde los usuarios podrán apuntarse de manera voluntaria. Estas actividades constarán de información como: número de vacantes disponibles, nombre de la actividad, donde se desarrolla y el horario. Un usuario será capaz de apuntarse e incluso de desapuntarse de las actividades, además estas actividades podrán ser actividades periódicas o no.

- **Todo tiene que estar en modo de lectura fácil**

Toda la página web, tanto la descripción, actividades, noticias, etc., tendrán que tener el modo de lectura fácil. Esto con el objetivo de facilitar a los usuarios el manejo de la página. Para ello también se colocarán imágenes en todas las actividades a modo descriptivo y orientativo sobre la actividad que se va a desarrollar por si los participantes no la conocen.



- **Notificar a los usuarios de las actividades**

Cuando quede un cierto periodo de tiempo, se le enviará un correo al usuario indicándole que queda poco tiempo para la realización de la actividad, esto es por si el usuario por alguna razón ha olvidado que se ha apuntado a una actividad. También se les dará la opción en dicho correo de acceder a la página para desapuntarse de la actividad en caso de que no puedan asistir. También se informará periódicamente a los usuarios sobre nuevas actividades o cursos que se hayan creados nuevos.

- **Lista de espera**

Todas las actividades están abiertas para todo tipo de usuarios, así bien, si la capacidad máxima de usuarios en la actividad es alcanzada, los usuarios que se apunten después entrarán automáticamente en una lista de espera para que en caso de que aluna persona se desapunte, enviar inmediatamente al siguiente en la lista de espera a ver si aún sigue interesado en asistir a la actividad.

- **Crear formularios para la valoración de las actividades**

Estos formularios irán destinados a los usuarios para puntuar las actividades a las que han asistido, tendrán la opción de puntuar la actividad y de dejar un pequeño comentario, de esta manera un usuario antes de apuntarse a una actividad, podrá ver las valoraciones y decidir.

- **Distintos programas**

Todas las actividades estarán dentro de distintos programas que se desarrollarán durante el año.

- **Los Profesionales podrán ver una lista de todos los usuarios registrados en la página web**

Esto se hace para poder llevar un control sobre los usuarios que hay en la página.

- **Diseño corporativo**

La página dispondrá de un diseño basado en el logo de la asociación, sobre todo verde. No será muy sobrecargada de elementos y será de fácil intuición de fácil uso.

- **Vista para el usuario**

Cuando un usuario inicia sesión, le aparecerá por pantalla todas las actividades que tiene los próximos siete días, también podría buscar actividades a lo largo del tiempo y podrá ver toda la información de la actividad que escoja.

- **Google Calendar**

La funcionalidad de poder añadir las actividades al Google calendar de cada participante, para facilitar a los usuarios el manejo de las actividades que tienen pendientes.

Estos son los requisitos funcionales que el cliente ha solicitado, además de los descritos se está trabajando en plantear nuevas funcionalidades para facilitar el trabajo del cliente. Como puede ser una herramienta de ayuda didáctica e interactiva en la navegación dentro de la página web, a modo de guía dentro del funcionamiento de la página web, además de la integración del modo oscuro de la propia aplicación, todo esto conservando los colores corporativos de la misma.

### 3.3. Identificación de los actores.

Tras una reflexión sobre los actores que debe tener esta aplicación, se ha llegado a la conclusión de que hay que aumentar los actores a: Nominal, Monitor, Profesional. Esto debido a que la segregación de funcionalidades es conveniente y sobre todo porque entregar la gestión de la funcionalidad entera a varias personas podría ser un problema a largo plazo, en el apartado siguiente se detallarán todas las funciones que va a tener cada actor.

#### 3.3.1. Usuario nominal

Este rol lo podrán hacer uso aquellas personas que van a hacer uso de la aplicación en sí, va dirigido a los integrantes de la asociación que quieran tener simplemente un uso didáctico de la aplicación, este usuario tendrá control sobre:

- Iniciar sesión dentro de la plataforma, ya sea mediante autenticación de Google (Larry Page & Sergey Brin, 1998) o mediante la ayuda de un *usuario monitor* o *usuario administrador*.
- Revisar su perfil creado con anterioridad y accesibilidad a editar los campos menos sensibles dentro del perfil, estos campos están debidamente marcados para el fácil reconocimiento del usuario.
- Podrá informarse sobre las actividades que están en curso, además podrá revisar los comentarios ingresados por asistentes sobre dicha actividad para que pueda decidir si apuntarse a la actividad o no. Finalmente se podrá desapuntar de la actividad si lo ve conveniente.
- En el mismo apartado podrá revisar toda la información relevante de la actividad, como el lugar de realización, la fecha cuando empezó y cuando termina. Para tener una visión global de la actividad, ésta misma tiene un apartado donde se hace una media de las notas que han puesto los asistentes y el total de comentarios añadidos.
- En otro apartado podrá revisar cuando tiene las próximas actividades a las que tiene que acudir, inicialmente en un plazo del día en concreto a siete días en adelante, el usuario si quiere puede quitar el filtro para poder ver todas las actividades a las que se ha ido apuntando.
- Para cada asistencia a la actividad el usuario podrá dejar una valoración de la actividad, la

valoración consta de una nota del 0 al 10 y un breve comentario, ambos campos obligatorios. Estas son las funcionalidades de las cuales podrá hacer uso un usuario Nominal.



*Imagen 14 Usuario Nominal*

### 3.3.2. Usuario Monitor

Se ha decidido implementar un nuevo actor, llamado Monitor, debido a que se ha visto necesario un papel el cual gestione las actividades y los horarios dentro de la aplicación, pero no tenga control total de la aplicación y su funcionamiento.

Además de poseer las funcionalidades para un usuario Nominal, tiene sus propias funcionalidades adicionales:

- Poder crear perfiles para nuevas incorporaciones a la aplicación, esto va dirigido sobre todo para los usuarios nominales, ya que ellos no se pueden registrar en la aplicación, si bien pueden acceder mediante su cuenta de Google, pero no pueden crearse su perfil dentro de la aplicación.
- Poder eliminar perfiles dentro de la aplicación esto para mantener una limpieza de perfiles ya no activos dentro de la aplicación.
- Poder crear horarios nuevos eligiendo la actividad correspondiente, la fecha de realización, la frecuencia, el horario de la actividad, la campaña a la que pertenece y el aforo informativo de dicha actividad.
- Poder eliminar horarios, esto sólo si ningún usuario está apuntado a dicha actividad o si la actividad ya ha pasado su fecha límite.
- Tener acceso a la lista de usuarios que están registrados en la aplicación.

Estas son las funcionalidades que puede realizar el usuario *Monitor*.



*Imagen 15 Usuario Monitor*

### 3.3.3. Usuario Administrador

Se ha tenido en mente la administración de la aplicación a un nivel mas profundo, para ello surge la idea del usuario *Administrador*, dicho usuario tendrá control de la aplicación en su totalidad, ya que, además de poseer las funcionalidades tanto del usuario *Nominal* como del usuario *Monitor*, posee algunas funcionalidades, sobre todo de gestión.

En el siguiente apartado se describen las funcionalidades del usuario *Administrador*:

- Obtener el listado de actividades, para con ello poder agregar nuevas actividades, para que estén disponibles para asignarlas a horarios, además de poder editar las mismas e incluso borrar las que no estén asignadas a un horario previamente.
- Obtener el listado de campañas, para con ello poder agregar nuevas campañas, para que estén disponibles para asignarlas al horario en cuestión, además de poder editar las mismas e incluso borrar las que no estén asignadas a un horario previamente.
- Obtener el listado de Tipos de Actividades, para con ello poder agregar Tipos de actividades dentro de los horarios, además de poder editar las existentes e incluso borrar las que no están previamente asignadas a un horario.
- Tener acceso al *Panel de Control*, donde se almacena toda la información importante de la aplicación.

Estas son las funcionalidades que tiene el usuario *Administrador*, por supuesto juega un papel más administrativo y con ello un papel con más responsabilidad. Dirigido a los monitores con mas experiencia.



Imagen 16 Usuario Administrador

## 4. Arquitectura del sistema.

Los diagramas a continuación representan la evolución de la definición de la base de datos y sus correspondientes diagramas.

### 4.1. Diagrama Entidad Relación Previo.

Se puede apreciar la falta de gestión de los archivos insertados por el usuario, se han mantenido la mayor parte de las tablas, también se ha prescindido de las tablas de *Profesional* y *Participante*, ya que se ha hecho una refactorización a las funcionalidades que se tenían en cuenta en el momento en el que se creó este diagrama. En el diagrama actualizado se explica la estructura con más detenimiento. Referencia a *Diagrama 1 Entidad Relación Previo*

Se ha descubierto posibles campos que podrían llegar a ser nulos y no debería ser así, teniendo en cuenta estos fallos en el diseño de la arquitectura de la base de datos, se ha procedido a realizar una nueva gestión de la misma.

### 4.2. Diagrama Entidad Relación Actualizado.

El diseño de esta estructura para la base de datos ha ido de la mano con las necesidades del proyecto, con el fin de desarrollar una estructura sólida y fácil de entender, teniendo en cuenta su complejidad añadida. Referencia a *Diagrama 2 Entidad Relación Actual*

Como una de las necesidades es que sea de lectura fácil, se ha tenido muy en cuenta el almacenamiento de imágenes por parte de cualquier tipo de usuario, principalmente a la hora de ilustrar una actividad y a la hora de seleccionar la foto de perfil de usuario.

Se explica con mucha más profundidad en el apartado, referencia a 4.4 *Diccionario de datos* más adelante.

### 4.3. Diagrama Relacionable Previo.

En este diagrama se puede apreciar, referencia a *Diagrama 3 Relacionable Previo*, con más profundidad los cambios realizados en el diseño, sobre todo a cómo se manejaba la gestión de las actividades (*tabla USUCALENDARIO*) con los usuarios, ya que no se disponía de la funcionalidad que permite al usuario dejar un comentario y una nota a la actividad a la cual ha asistido.

Por otra parte, se ha hecho una edición de campos en la tabla (*ACTIVIDAD*), quitando campos y relaciones con las tablas (*CAMPAÑA*, *TIPO*), ya que se ha llegado a la conclusión de que puede haber la misma actividad, pero en varios horarios, y no necesariamente dicha actividad va a pertenecer a la misma campaña o tipo siempre, de esta manera se da más juego a la variedad de actividades y reutilización de las mismas. Referencia a *Diagrama 4 Relacionable Actual*

La tabla que pasa a ser la protagonista es (*HORARIO*), ya que dicha tabla albergará toda la información necesaria, brindando al *MONITOR O ADMINISTRADOR*, la capacidad de crear todo tipo de horarios con la libertad de elegir tanto la actividad, como el tipo y finalmente la campaña a la que pertenece.

Esto es todo lo referente a las diferencias entre la arquitectura de la base de datos del diseño antiguo

con la arquitectura que se ha llevado a cabo a lo largo del desarrollo del proyecto.

#### 4.4. Diccionario de datos.

A continuación, se procede a explicar cada una de las tablas que se han implementado en el desarrollo, además de explicar cada uno de sus atributos. Cabe destacar que se ha decidido cambiar el tipo de dato de todas las claves primarias a un campo GUID. Referencia a *Diagrama 4 Relacionable Actual*

Un tipo de dato GUID (*Globally Unique Identifier*) es un identificador único, conformado por 128 bits (16 bytes) que se suelen representar como una cadena de 32 caracteres hexadecimales, separados por guiones en 5 grupos, lo que significa que se generan identificadores únicos en sistemas sin riesgo de colisiones. Con esto evitando la duplicidad de claves primarias entre registros de la misma tabla.



Imagen 17 Ejemplo GUID

Teniendo en cuenta esta breve explicación, se procede a explicar las tablas:

- **Tabla Usuario:**

Esta tabla contendrá la información de un usuario registrado en la aplicación, cabe destacar que debido a que se permite la autenticación con Google, solo existen cuatro campos obligatorios dentro de esta tabla, sin contar con el identificador único. Estos campos son: *Email*, *Nombre*, *Primer Apellido* y *Foto* (*almacenamiento*). Esto es debido a que es la información que provee Google sobre los perfiles.

Explicación de los campos:

**ID\_usuario:** será la clave primaria de cada registro, está conformado por un identificador GUID. Por lo tanto, no puede ser nulo.

**DNI:** contendrá la información del documento de identidad del usuario, ya sea NIF o DNI, con una máxima longitud de 9 caracteres, VARCHAR(9). Este campo puede ser nulo debido a que los usuarios que se registran a la aplicación mediante la autenticación de Google, inicialmente solo rellenan los campos de Correo, Nombre, Primer Apellido e imagen. Posteriormente podrán editar su correspondiente campo. Finalmente, este campo es una clave de la tabla, lo que quiere decir que no se puede repetir más de un registro con el mismo DNI.

**DNI\_tutor:** este campo contendrá el DNI del tutor al que está asignado el usuario, VARCHAR(9), este campo puede ser nulo debido a que no todos los usuarios tendrán por que tener un tutor asociado.



**ID\_almacenamiento:** este campo contendrá el identificador que asocia la imagen del usuario dentro de la tabla de metadatos en el almacenamiento, por lo tanto, es un campo de tipo GUID, debido a que se trata de una clave foránea, además este campo no puede ser nulo.

**Rol:** este campo se trata de un ENUM, cuya función es informar sobre el rol que va a ejercer el usuario, el rol podrá ser Nominal, Monitor y Administrador. Este campo no puede ser nulo.

**Nombre:** este campo contendrá el nombre de pila del usuario de máximo de 50 caracteres, VARCHAR(50), este campo no podrá ser nulo.

**Apellido\_1:** este campo contendrá el primer apellido del usuario de máximo de 50 caracteres, VARCHAR(50), este campo no podrá ser nulo.

**Apellido\_2:** este campo contendrá el segundo apellido del usuario con un máximo de 50 caracteres, VARCHAR(50), éste campo puede ser nulo debido a que no proviene en los datos enviados para aquellos usuarios que se registren mediante Google.

**Email:** este campo contiene la dirección de correo del usuario, con formato de correo electrónico, VARCHAR(50). Se trata de un campo KEY, lo que significa que no pueden existir registros con el mismo Email dentro de la tabla, el campo no puede ser nulo.

**Contrasenya:** este campo contiene la contraseña del usuario, el usuario ingresará una contraseña de mínimo 5 caracteres y dentro de la base de datos esta contraseña estará encriptada, para mejorar la seguridad del usuario por lo que el campo tendrá que ser un campo VARCHAR(255). Este campo no puede ser nulo.

**Dirección:** este campo contiene información sobre la vivienda del usuario, por tema de confidencialidad de datos se permite al usuario ingresar una dirección de hasta máximo 60 caracteres, VARCHAR(60). Esté campo puede ser nulo.

**Teléfono:** este campo contiene información sobre el teléfono de contacto del usuario, se trata de un campo VARCHAR(9), esto es debido a que la aplicación se desarrolla dentro de España, pero en un futuro se podría agregar las terminaciones de distintos números según el país. Este campo puede ser nulo.

**FechaNacimiento:** este campo contiene información sobre la fecha de nacimiento del usuario, tipo DATE. Este campo podrá ser nulo. Y podrá ser actualizado una vez solo para aquellos usuarios que hayan iniciado sesión mediante Google.

- **Tabla Horario**

Esta tabla representa toda la información necesaria para plasmar el desarrollo de una actividad, contendrá campos necesarios para la identificación de la misma dentro de un periodo de tiempo.

Explicación de los campos:

**ID\_horario:** será la clave primaria de cada registro, está conformado por un identificador GUID. Por lo tanto, no puede ser nulo.

**ID\_actividad:** será la clave foránea, contiene la referencia al registro de la tabla Actividad. No puede ser nulo.

**ID\_tipo:** será la clave foránea, contiene la referencia al registro de la tabla Tipo. No Puede ser nulo.

**ID\_campaña:** será la clave foránea, contiene la referencia al registro de la tabla Campaña. No puede ser nulo.

**Dirección:** este campo contiene información sobre la situación donde se va a desarrollar la actividad, es aconsejable que se introduzca la dirección de Google Maps, pero también se puede añadir una dirección de hasta 70 caracteres, VARCHAR(70), este campo no puede ser nulo.

**DiaSemana:** Este campo representa el día de la semana en el cual se va a desarrollar la actividad dentro de este horario, es un campo que admite una cadena de texto, VARCHAR(20), este campo no puede ser nulo.

**HoraInicio:** este campo representa la hora de inicio de la actividad, tipo TIME, con el formato HH:MM, este campo no puede ser nulo.

**HoraFinal:** este campo representa la hora de finalización de la actividad, tipo TIME, con el formato HH:MM, este campo no puede ser nulo.

**Frecuencia:** este campo representa el tipo de actividad que se va a desarrollar, se trata de un tipo ENUM, y permite elegir entre *Puntual* y *Semanal*.

**FechaInicio:** este campo representa la fecha de inicio de la actividad, de tipo Date y con el formato YYYY/MM/DD, este campo no puede ser nulo.

**FechaFin:** este campo representa la fecha de finalización de la actividad, de tipo Date y con el formato YYYY/MM/DD, este campo puede ser nulo, ya que las actividades que sean de tipo *Puntual* solo van a tener fecha de inicio.

**Aforo:** este campo representa la cantidad de usuarios recomendada para esta actividad, en un principio se pensó en limitar las actividades, pero ya que se trata de una asociación donde se intenta fomentar la inclusión de personas, no conviene limitar las asistencias a las actividades. Este campo no puede ser nulo, de tipo TinyInt.

**Ocupación:** este campo va de la mano con el anterior, representa la cantidad de usuarios que han asistido a la actividad, para tener un registro sobre la asistencia a la actividad, es de tipo TinyInt, puede ser nulo.



- **Tabla Horario\_Usuario**

Se trata de una tabla intermedia entre las tablas *Usuario* y *Horario*, tiene la función de llevar un historial sobre a qué horarios ha asistido un usuario, así como la puntuación seleccionada y el comentario dejado por el usuario.

Explicación de los campos:

**ID\_usuario\_horario:** será la clave primaria de cada registro, está conformado por un identificador GUID. Por lo tanto, no puede ser nulo.

**ID\_usuario:** será la clave foránea, contiene la referencia al registro de la tabla Usuario. No puede ser nulo.

**ID\_horario:** será la clave foránea, contiene la referencia al registro de la tabla horario. No puede ser nulo.

**Comentario:** este campo representa el pequeño comentario que puede dejar el usuario sobre su experiencia con la actividad, o puede dejar un comentario sobre lo que espera de la actividad, tipo VARCHAR(50). Puede ser nulo.

**Nota:** este campo representa la calificación por parte del usuario hacia la actividad en el horario, puede ir del 1 al 10. Tipo TinyInt, puede ser nulo.

**FechaDeAsistencia:** este campo indica la fecha en la que tendrá que asistir el usuario a dicha actividad. Tipo Date, no puede ser nulo.

El funcionamiento de esta tabla se ha llevado a cabo la idea de tener actividades periodicas y actividades puntuales. Cada vez que un usuario se registra en una actividad ocurre una gestión por detrás.

Si la actividad es puntual solamente se genera un registro en esta tabla con la fecha de desarrollo de la actividad. Por el contrario, si la actividad es semanal, se generan tantos registros como semanas haya de por medio entre la fecha de inicio de la actividad y la fecha de finalización, cada una de las asistencias con su correspondiente puntuación y comentario, cabe destacar que solo se generan registros a partir de la fecha en la que el usuario se encuentra.



- **Tabla Tipo**

Esta tabla representa los tipos de actividades que se pueden desarrollar, como, por ejemplo: virtual, exterior, interior, etc.

Explicación de los campos:

**ID\_tipo:** será la clave primaria de cada registro, está conformado por un identificador GUID. Por lo tanto, no puede ser nulo.

**NombreTipo:** este campo representará del nombre del tipo, tipo VARCHAR(30), no puede ser nulo.

**DescripciónTipo:** este campo representará la descripción del tipo, para agregar información a la misma, tipo VARCHAR(60), no puede ser nulo.

- **Tabla Campanya**

Esta tabla representa las distintas campañas a las cuales puede pertenecer la actividad, a petición del cliente.

Explicación de los campos:

**ID\_campaña:** será la clave primaria de cada registro, está conformado por un identificador GUID. Por lo tanto, no puede ser nulo.

**NombreCampaña:** este campo representará del nombre de la campaña, tipo VARCHAR(30), no puede ser nulo.

**DescripciónCampaña:** este campo representará la descripción de la campaña, para agregar información a la misma, tipo VARCHAR(60), no puede ser nulo.

**Fechalnicio:** este campo representa la fecha de inicio de la campaña, de tipo Date y con el formato YYYY/MM/DD, este campo no puede ser nulo.

**FechaFin:** este campo representa la fecha de finalización de la campaña, de tipo Date y con el formato YYYY/MM/DD, este campo no puede ser nulo.

- **Tabla Actividad**

Esta tabla representa la actividad en sí que se va a desarrollar, no pertenece a ningún tipo ni a ningún programa, de esa manera esa actividad se puede realizar en distintas campañas y con distintos tipos, todo esto sucede en la tabla de horario. Con la finalidad de brindar al Monitor o administrador las herramientas necesarias para que pueda crear todo tipo de actividades dentro de la aplicación.

Explicación de los campos:

**ID\_actividad:** será la clave primaria de cada registro, está conformado por un identificador GUID. Por lo tanto, no puede ser nulo.

**NombreActividad:** este campo representará del nombre de la actividad, tipo VARCHAR(30), no puede ser nulo.

**DescripciónActividad:** este campo representará la descripción de la actividad, para agregar información a la misma, tipo VARCHAR(60), no puede ser nulo.

**ID\_Almacenamiento:** será la clave foránea, contiene la referencia al registro de la tabla Almacenamiento donde se encuentra la foto de la actividad. Es en aquella tabla donde se guardan todas las imágenes introducidas en la aplicación. Todo esto para facilitar el entendimiento de la misma. Este campo no puede ser nulo.

- **Tabla Almacenamiento**

La función de esta tabla es guardar toda la meta información sobre las imágenes introducidas en la aplicación, ya sean por parte del usuario (fotos de perfil) como por actividades (foto ilustrativa de la actividad).

**ID\_Almacenamiento:** será la clave primaria de cada registro, está conformado por un identificador GUID. Por lo tanto, no puede ser nulo.

**NombreArchivo:** este campo representará del nombre del archivo, tipo VARCHAR(30), no puede ser nulo.

**url:** este campo nos muestra la dirección donde se encuentra alojado el archivo, para poder acceder a él desde la aplicación. Tipo VARCHAR(255) no puede ser nulo.

Cabe destacar que, aunque los usuarios sean capaces de subir archivos a la aplicación, estos solo podrán ser archivos con un formato de imagen, no se les permite subir archivos de texto, video o presentación.

#### 4.5. Explicación de las relaciones entre tablas.

##### **Relación entre Usuario y Horario**

Entre estas tablas existe una relación **N,N**, debido a que un usuario puede asistir a varios horarios y un horario puede tener varios usuarios apuntados. Esto quiere decir que, tras la relación de estas dos tablas, nace una tabla intermedia *Usuario\_Horario*, esta tabla contendrá las claves primarias de ambas tablas que une, además estas claves foráneas serán la clave primaria conjunta de la tabla.

##### **Relación entre Horario y Actividad**

La relación presente entre estas dos tablas es una **1,N**, debido a que un horario solo puede tener asignada una actividad, mientras que una actividad puede estar presente en varios horarios.

##### **Relación entre Horario y Tipo**

La relación presente entre estas tablas es una **1,N**, debido a que un horario solo puede pertenecer a un tipo, mientras que un tipo puede estar asignado a varios horarios.

##### **Relación entre Horario y Campaña**

La relación presente entre estas tablas es una **1,N**, debido a que un horario solo puede pertenecer a una campaña, mientras que una campaña puede tener varios horarios asignados.

##### **Relación entre Almacenamiento y Usuario**

La relación entre estas dos tablas es una **1,N**, debido a que un usuario puede tener varias fotos asignadas a lo largo de l tiempo y una foto solo puede estar asignada a un usuario.

##### **Relación entre Almacenamiento y Actividad**

La relación entre estas dos tablas es una **1,N**, debido a que una actividad puede tener varias fotos asignadas a lo largo del tiempo y una foto solo puede estar asignada a una actividad.

##### **Relación entre Usuario y Usuario**

Entre la propia tabla usuario existe una relación reflexiva, debido a que un usuario puede tener un usuario asignado como tutor.

## 5. Diseño e implementación del sistema.

### 5.1. Diseño del servidor.

Empezando por las tecnologías utilizadas para desarrollar este punto, junto con nodeJs se ha usado frameworks como express (TJ Holowaychuk, 2012), el cual es un framework minimalista y flexible que facilita la creación de APIs y aplicaciones web. Con esto agilizando la creación de la estructura de la API.

Se ha utilizado un ORM llamado Sequelize (*Sequelize | Feature-Rich ORM for Modern TypeScript & JavaScript*, n.d.), un ORM actúa como una capa intermedia entre la aplicación y la base de datos, abstrayendo los detalles de implementación de la base de datos en este proyecto.

Esta herramienta nos permite generar nuestras propias entidades, con sus respectivos atributos y relaciones entre tablas, estas se crean automáticamente en la base de datos una vez se ejecute el programa.

#### 5.1.1. Estructura de carpetas.

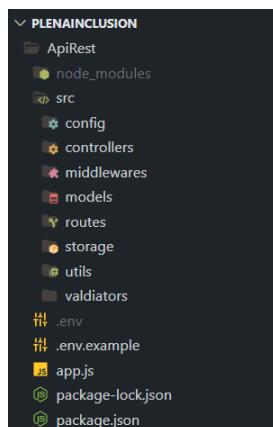
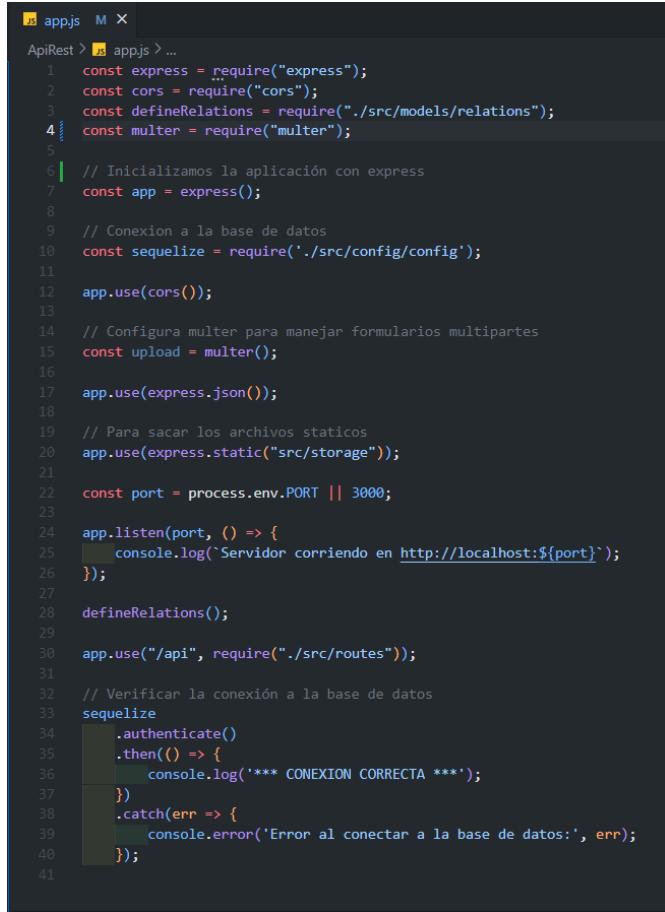


Imagen 18 Estructura de Carpetas en el Servidor

El proyecto se encuentra en la carpeta “*ApiRes*”, esta carpeta está constituida por el archivo principal *app.js*, la carpeta “*src*”, las variables de entorno “*.env*” y las dependencias del proyecto en la carpeta “*node\_Modules*” junto con la configuración del mismo en el fichero “*package.json*”.

Dentro de la carpeta “*src*” se puede observar que hay un grupo de carpetas, cada carpeta contiene una capa de la aplicación, todo esto para poder ajustarnos a las reglas REST, para hacer de la API lo más segura y escalable posible.

### 5.1.2. Archivo principal app.js.



```

app.js M X
ApiRest > app.js > ...
1 const express = require("express");
2 const cors = require("cors");
3 const defineRelations = require("./src/models/relations");
4 const multer = require("multer");
5
6 // Inicializamos la aplicación con express
7 const app = express();
8
9 // Conexión a la base de datos
10 const sequelize = require('./src/config/config');
11
12 app.use(cors());
13
14 // Configura multer para manejar formularios multipartes
15 const upload = multer();
16
17 app.use(express.json());
18
19 // Para sacar los archivos staticos
20 app.use(express.static("src/storage"));
21
22 const port = process.env.PORT || 3000;
23
24 app.listen(port, () => {
25   console.log(`Servidor corriendo en http://localhost:\${port}`);
26 });
27
28 defineRelations();
29
30 app.use("/api", require("./src/routes"));
31
32 // Verificar la conexión a la base de datos
33 sequelize
34   .authenticate()
35   .then(() => {
36     console.log('*** CONEXIÓN CORRECTA ***');
37   })
38   .catch(err => {
39     console.error('Error al conectar a la base de datos:', err);
40   });
41

```

Imagen 19 app.js

Este es el archivo principal de la aplicación, se puede apreciar que en las primeras líneas importamos todo lo necesario para que este archivo pueda ser lo más resumido posible.

Primero inicializa la aplicación utilizando “Express”, seguidamente se hace la conexión con la base de datos utilizando “Sequelize”, esto nos permitirá establecer nuestras tablas y nuestras relaciones entre ellas.

Seguidamente se configura las CORS (Cross-Origin Resource Sharing), es decir, se habilita la accesibilidad a la API desde distintos dominios, esto es crucial para el desarrollo de aplicaciones web donde tiene que existir una comunicación con el cliente (en este caso en Angular) y el servidor.

Se configura la manera de recibir datos por parte de la API, en este caso se han configurado dos opciones, como se puede ver, la primera es con “multer”, el cual es un middleware que facilita la gestión de archivos en solicitudes “*multipart/form-data*”. Estas solicitudes se envían normalmente para enviar varios tipos de datos, mediante clave valor, así se pueden transmitir varios datos de varios tipos en una sola llamada (en este caso se usa para poder almacenar correctamente las

imágenes).

La segunda manera en la que se configura la manera de recibir los datos es mediante solicitudes JSON, es la manera más estándar de transmitir datos entre el cliente y el servidor, mediante el uso de objetos transformados a un formato JSON. Finalmente se configura el lugar de almacenamiento donde se van a guardar todos los datos que provengan de una solicitud “Multipar/Form-Data”.

Se inicia el puerto donde el servidor va a estar recibiendo todas las solicitudes e intercambiando información. Seguidamente se definen todas las relaciones de las tablas que se han definido con anterioridad.

Por último se declara la carpeta donde van todas las rutas de la aplicación, dentro de la carpeta “routes”, y se manda un mensaje por la consola para comunicar que todo ha sucedido de manera correcta.

#### 5.1.3. Variables de entorno.

```
ApiRest > cat .env
1 PORT=3000
2 DATABASEIP=localhost
3 DATABASEPORT=3306
4 DATABASEUSER=root
5 DATABASEPASS=
6 PUBLIC_URL=http://localhost:3000
7 JWT_SECRET=LlaveMaestra
```

Imagen 20 Variables de Entorno

Este archivo se encarga de almacenar datos importantes como se pueden ver en la imagen, un ejemplo son los datos de conexión con la base de datos. Es buena *práctica* debido a que son datos sensibles que no pueden ser expuestos con facilidad, normalmente este archivo no se sube al repositorio de archivos y esto garantiza que la aplicación sea *más* robusta y que los datos del usuario estén protegidos.

Un ejemplo perfecto es la definición del “JWT\_SECRET”, ya que esta es la manera de generar los tokens, con los cuales el cliente podrá iniciar sesión y mantenerla iniciada. Son datos que no se pueden compartir de manera deliberada y mediante esta herramienta se pueden seguir usando y llamando dentro de la aplicación.

#### 5.1.4. Archivo de configuración con la base de datos.



```
config.js M ×
ApiRest > src > config > config.js ...
1 const Sequelize = require("sequelize");
2 require("dotenv").config();
3
4 // Realiza la conexión con la base de datos, enviando un objeto con
5 // los datos necesarios para establecer la conexión.
6 const sequelize = new Sequelize('tfq_plena_inclusion',
7   process.env.DATABASEUSER,
8   process.env.DATABASEPASS, {
9     host: process.env.DATABASEIP,
10    dialect: 'mysql'
11  });
12
13 module.exports = sequelize;
```

Imagen 21 Configuración Base de Datos

En este archivo se define la conexión con la base de datos mediante “Sequelize”, de una manera bastante sencilla y eficaz, para ello se necesita las configuraciones necesarias mediante las cuales (simplemente cambiando las variables de entorno) se podría conectar a cualquier tipo de base de datos, ya sea alojada en local o en la nube.

#### 5.1.5. Carpeta Models.

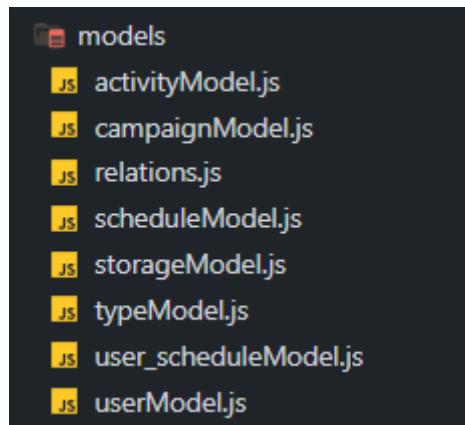


Imagen 22 Carpeta Models

En esta carpeta es la capa donde se almacenan todos los modelos que se van a requerir en la construcción de la base de datos, por supuesto cada uno de estos modelos corresponden a los descritos anteriormente en la explicación de la base de datos.

Estos modelos, una vez que el archivo principal se ejecute, migrarán su información de manera que el esquema de base de datos que esté configurado, recibirá estos modelos generará la base de datos con sus respectivas relaciones de manera inmediata.

A continuación, se presenta un ejemplo de cómo se genera un modelo, en este caso será el de usuario:



```

1  const { DataTypes } = require('sequelize');
2  const db = require('../config/config.js');
3
4  // Se define el modelo del usuario dentro de la base de datos.
5  const UserModel = db.define('User', {
6    ID_user: {
7      type: DataTypes.UUID,
8      defaultValue: DataTypes.UUIDV4,
9      primaryKey: true
10   },
11   DNI: {
12     type: DataTypes.STRING,
13     allowNull: true,
14     unique: true
15   },
16   Rol: {
17     type: DataTypes.ENUM('Nominal', 'Monitor', 'Administrador'),
18     allowNull: false
19   },
20   Name: {
21     type: DataTypes.STRING,
22     allowNull: false
23   },
24   Surname_1: {
25     type: DataTypes.STRING,
26     allowNull: false
27   },
28   Surname_2: {
29     type: DataTypes.STRING,
30     allowNull: true
31   },
32   Email: {
33     type: DataTypes.STRING,
34     allowNull: false,
35     unique: true
36   },
37   Pass: {
38     type: DataTypes.STRING,
39     allowNull: true,
40     select: false
41   }
42 });

```

Imagen 23 Modelo Usuario

En este archivo se puede apreciar que, mediante la configuración de la base de datos, se puede crear una tabla de manera sencilla, configurando los campos de la misma y los tipos de datos que va a contener, así como sus claves primarias y claves. Se puede apreciar en el campo de Email se hace uso de “Unique”, esto hace que ese campo sea una clave y que no se puedan repetir varios registros con el mismo campo.



```

}, {
  freezeTableName: true,
  timestamps: true,
  tableName: 'User',
  paranoid: true,
  deletedAt: 'softDelete'
});

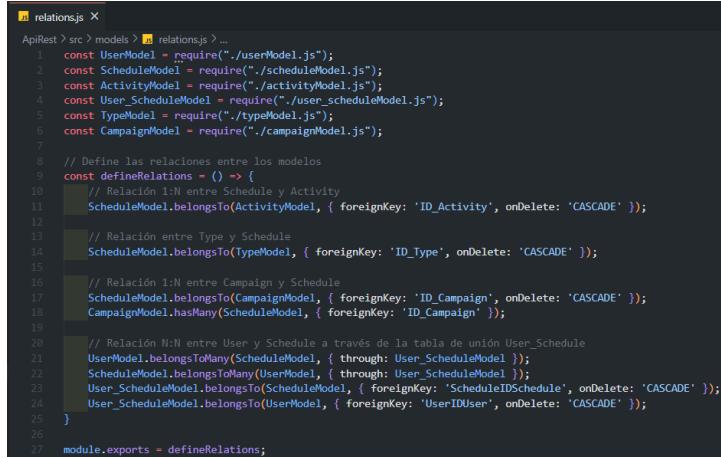
(async () => {
  await db.sync();
})();

module.exports = UserModel;

```

Imagen 24 Soft Delete

Algo a destacar dentro de la construcción de la base de datos, es el concepto de “soft-delete”, que es una técnica en la que los registros no se eliminan físicamente de la base de datos, sino que se marcan como eliminados mediante un campo específico añadido (deletedAt en este caso), permitiendo que los datos sean recuperables y manteniendo la integridad histórica. Esto hace que se pueda brindar al cliente una funcionalidad extra, la cual es la capacidad de poder recuperar datos previamente borrados.



```
relations.js
1 const UserModel = require("./userModel.js");
2 const ScheduleModel = require("./scheduleModel.js");
3 const ActivityModel = require("./activityModel.js");
4 const UserScheduleModel = require("./user_scheduleModel.js");
5 const TypeModel = require("./typeModel.js");
6 const CampaignModel = require("./campaignModel.js");
7
8 // Define las relaciones entre los modelos
9 const defineRelations = () => {
10   // Relación 1:N entre Schedule y Activity
11   ScheduleModel.belongsTo(ActivityModel, { foreignKey: 'ID_Activity', onDelete: 'CASCADE' });
12
13   // Relación entre Type y Schedule
14   ScheduleModel.belongsTo(TypeModel, { foreignKey: 'ID_Type', onDelete: 'CASCADE' });
15
16   // Relación 1:N entre Campaign y Schedule
17   ScheduleModel.belongsTo(CampaignModel, { foreignKey: 'ID_Campaign', onDelete: 'CASCADE' });
18   CampaignModel.hasMany(ScheduleModel, { foreignKey: 'ID_Campaign' });
19
20   // Relación N:N entre User y Schedule a través de la tabla de unión User_Schedule
21   UserModel.belongsToMany(ScheduleModel, { through: User_ScheduleModel });
22   ScheduleModel.belongsToMany(UserModel, { through: User_ScheduleModel });
23   User_ScheduleModel.belongsTo(ScheduleModel, { foreignKey: 'ScheduleIDSchedule', onDelete: 'CASCADE' });
24   User_ScheduleModel.belongsTo(UserModel, { foreignKey: 'UserIDUser', onDelete: 'CASCADE' });
25 }
26
27 module.exports = defineRelations;
```

Imagen 25 Archivo de Relaciones

En este archivo se definen todas las relaciones que van a tener las tablas, estas relaciones van de acuerdo a las relaciones previamente descritas en la explicación de la arquitectura de datos.

#### 5.1.6. Carpeta Routes.

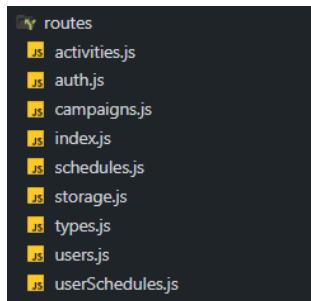


Imagen 26 Carpeta Routes

En esta carpeta se almacenan todas las rutas (end-points) de la aplicación, se ha configurado de tal manera de que cada familia de ruta tenga sus verbos en su totalidad, garantizando así la estructura REST, estos verbos son GET (se utiliza para la solicitud de datos), POST (se utiliza para la inserción de datos), PUT (se utiliza para la edición de datos) y DELETE (se utiliza para el borrado de datos)

A continuación, se muestra un ejemplo de la ruta de usuarios, donde se puede apreciar todo tipo de validaciones y autorizaciones en las rutas, garantizando así la integridad de los datos recibidos por parte de la API y la de la misma contra aquellas personas que intenten acceder a la misma



```

1  const express = require("express");
2  const router = express.Router();
3  const { getAllUsers, getUser, postUser, updateUser, deleteUser } = require("../controllers/userController");
4  const { validatorCreateUser, validator GetUser } = require("../validators/users")
5  const authMiddleware = require("../middlewares/session");
6  const uploadMiddleware = require("../utils/handleStorage");
7  const { checkRol } = require("../middlewares/rol");
8
9  // Ruta para listar todos los usuarios
10 router.get("/", authMiddleware, checkRol(['Monitor', 'Administrador']), getAllUsers);
11
12 // Ruta para recoger un usuario mediante su id
13 router.get("/:id", authMiddleware, checkRol(['Monitor', 'Administrador']), validator GetUser, getUser);
14
15 // Ruta para crear un usuario
16 router.post("/", authMiddleware, checkRol(['Monitor', 'Administrador']), validatorCreateUser, postUser);
17
18 // Ruta para modificar un usuario
19 router.put("/:id", authMiddleware, checkRol(['Monitor', 'Administrador']), uploadMiddleware.single("Photo"), updateUser);
20
21 // Ruta para eliminar un usuario.
22 router.delete("/:id", authMiddleware, checkRol(['Monitor', 'Administrador']), validator GetUser, deleteUser);
23
24 module.exports = router;

```

*Imagen 27 End Points de Usuario*

Se puede ver que las rutas en este caso respetarían las normas REST, además de ello, para garantizar un buen uso de la API por parte del usuario, se hace uso de varios middlewares.

El primero y más importante es el middleware de autenticación, este se encarga de verificar que el usuario que está tratando de acceder a la ruta está registrado en la aplicación y que además haya iniciado sesión.

Todo el sistema de autenticación se hace mediante JWT (Michael B. Jones & John Bradley, 2010), este es un estándar de tokens basado en JSON que se utiliza para transmitir información de forma segura entre dos partes, en esta ocasión utilizado para la autorización dentro de la aplicación. Esto quiere decir que, para hacer uso de todas las rutas de la aplicación, el usuario tiene que estar registrado y haber iniciado sesión.

Además, las rutas están protegidas a nivel de usuario, como se puede apreciar, se hace uso de un middleware, el cual se encarga de comprobar el rol del usuario que está tratando de realizar la solicitud, si coincide con los que están permitidos, la solicitud se realizará sin problemas, de lo contrario no será permitido el acceso al usuario.

También se puede apreciar un middleware que se encarga de la validación de datos que recibe el servidor por parte del cliente, si alguno de estos datos no coincide con los datos que debería recibir, enviará un error al cliente. Esto se explicará con más de talle en el apartado de Validadores.

Por último, se puede apreciar el middleware de *Multer*, el cual se encarga de gestionar los archivos enviados mediante una solicitud “multipart/form-data”, y guardarlos correctamente en la aplicación.

### 5.1.7. Carpeta Middlewares.

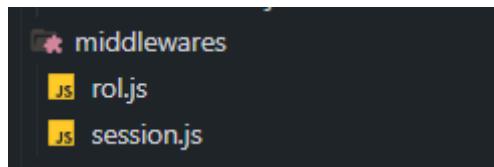


Imagen 28 Carpeta Middlewares

En esta carpeta se encuentran los middlewares utilizados en esta aplicación, sobre todo se enfocan en el control del inicio de sesión. Para ello se tiene un orden:



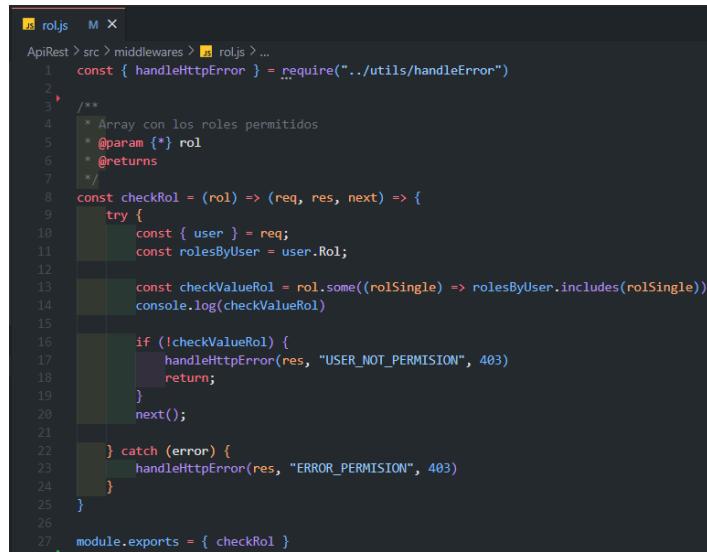
```
session.js M
ApiRest > src > middlewares > session.js > ...
1 const { handleHttpError } = require("../utils/handleError")
2 const { verifyToken } = require("../utils/handleJwt")
3 const UserModel = require("../models/userModel");
4
5 // Middleware para comprobar si la petición posee un token de sesión
6 // válido.
7 const authMiddleware = async (req, res, next) => {
8   try {
9     if (!req.headers.authorization) {
10       handleHttpError(res, "NOT-TOKEN", 401);
11       return;
12     }
13     const token = req.headers.authorization.split(' ').pop();
14     const dataToken = await verifyToken(token);
15
16     if (!dataToken.ID_user) {
17       handleHttpError(res, "ERROR_ID_TOKEN", 401);
18       return;
19     }
20
21     const user = await UserModel.findByPk(dataToken.ID_user);
22     req.user = user;
23
24     next();
25
26   } catch (error) {
27     handleHttpError(res, "NOT_SESSION", 401);
28     return;
29   }
30 }
31 module.exports = { authMiddleware };
32
```

Imagen 29 Token Middleware

En este archivo se puede apreciar cómo se intercepta la comunicación con la API, se recoge de las cabeceras el token enviado con la petición, se verifica si existe un token, si el token no es válido o no existe, no se permite acceder a la ruta. Al contrario, si se trata de un token valido se permite acceder a la ruta.

Esto ayuda a proteger las rutas de posibles ataques de terceros, permitiendo la comunicación con la API solo de usuarios autenticados.

Además, para agregar seguridad, se ha creado un middleware que sirve para identificar el rol del usuario tras su petición, así pues, se puede restringir la accesibilidad de las rutas también por parte de los usuarios registrados dentro de la aplicación.



```
rol.js M X
ApiRest > src > middlewares > rol.js > ...
1 const { handleHttpError } = require("../utils/handleError")
2
3 /**
4  * Array con los roles permitidos
5  * @param {*} rol
6  * @returns
7 */
8 const checkRol = (rol) => (req, res, next) => {
9   try {
10     const { user } = req;
11     const rolesByUser = user.Rol;
12
13     const checkValueRol = rol.some((rolSingle) => rolesByUser.includes(rolSingle))
14     console.log(checkValueRol)
15
16     if (!checkValueRol) {
17       handleHttpError(res, "USER_NOT_PERMISSION", 403)
18       return;
19     }
20     next();
21
22   } catch (error) {
23     handleHttpError(res, "ERROR_PERMISSION", 403)
24   }
25 }
26
27 module.exports = { checkRol }
```

Imagen 30 Middleware de Verificación de Rol

Como se puede apreciar en el archivo, este middleware intercepta la petición del usuario, recoge rol del usuario que ha lanzado la petición, y según los roles que recibe como parámetro, permite el acceso no lo restringe.

#### 5.1.8. Carpeta Controllers.

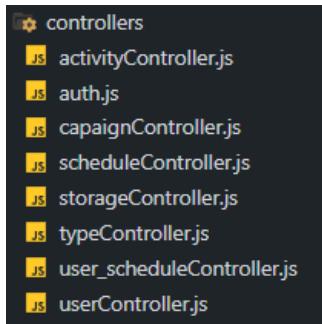


Imagen 31 Carpeta Controllers

Esta carpeta sirve como capa de interacción con la base de datos, cada archivo es llamado por su correspondiente ruta y en ella se accede al método necesario según el verbo de la petición, *GET*, *POST*, *PUT* o *DELETE*.

En cada uno de estos archivos se gestionaría el CRUD de cada modelo en la base de datos, garantizando así una buena división entre capas, cumpliendo así con la estructura REST.

En el siguiente apartado se procede a mostrar un ejemplo de cómo estaría formado un controlador.

```
/**  
 * Metodo que devuelve todos las actividades de la base de datos.  
 * @param {*} req  
 * @param {*} res  
 */  
const getAllActivities = async (req, res) => {  
    try {  
        const activities = await ActivityModel.findAll();  
        res.json(activities);  
    } catch (error) {  
        handleHttpError(res, "ERROR_GET_ACTIVITIES")  
    }  
}
```

Imagen 32 Controller GET

Este controlador se utiliza para listar todos los usuarios de la base de datos, se puede apreciar que se ha ideado un manejador de errores cuya funcionalidad es tratar los errores ocurridos en las peticiones de una manera más eficaz.

```
/**  
 * Metodo que devuelve una actividad por su identificador  
 * @param {*} req  
 * @param {*} res  
 */  
const getActivity = async (req, res) => {  
    try {  
        req = matchedData(req);  
        const { id } = req;  
        const activity = await ActivityModel.findByPk(id);  
        res.json(activity);  
    } catch (error) {  
        handleHttpError(res, "ERROR_GET_ACTIVITIES")  
    }  
}
```

Imagen 33 Controller GetByID

En este controlador se puede observar que en la petición se envía la clave primaria de la actividad a buscar, esta información es analizada mediante *Express*, con la herramienta de *matchedData*, esto lo que hace es comparar la información enviada en la petición con la información que el controlador recoge, tras analizar esto solamente recoge la información necesaria de la petición.

Tras buscar la actividad la devuelve en formato *JSON*, se puede observar el manejo de errores personalizado que se ha utilizado en este método también. Más adelante en el apartado útiles se explica con más detalle.

```

/*
 * Método que crea una nueva actividad en la base de datos
 * @param {*} req
 * @param {*} res
 */
const postActivity = async (req, res) => {
  try {
    const formData = req.body;
    const file = req.file;
    if (file) {
      formData.Photo = file.filename;
    }
    const activityData = {
      Name: formData.Name,
      Description: formData.Description,
      Photo: `http://${process.env.DATABASEIP}:${process.env.PORT}/${formData.Photo}`,
    }
    const data = await ActivityModel.create(activityData);
    res.status(201).json({ activityData });
  } catch (e) {
    handleHttpError(res, 'ERROR_POST_ACTIVITY')
  }
}


```

Imagen 34 Controller POST

Este controlador se encarga de insertar una actividad en la base de datos, el tratamiento de la información en este controlador es distinta, debido a que no es una solicitud de tipo JSON, sino que se trata de una solicitud *Multipart/FormData*. Esto hace que se tenga que hacer uso del middleware *Multer*. Para con ello poder insertar la imagen con la referencia a la actividad que pertenece.

Una vez insertada la imagen, se devuelve la URL donde se va a poder acceder a la imagen insertada por el usuario y por último se devuelve un mensaje de éxito al usuario si todo ha ido correctamente.

```

/*
 * Método que actualiza una actividad dado su identificador.
 * @param {*} req
 * @param {*} res
 * @returns
 */
const updateActivity = async (req, res) => {
  try {
    const { id } = req.params;
    const formData = req.body;
    const file = req.file;

    if (file) {
      formData.Photo = `http://${process.env.DATABASEIP}:${process.env.PORT}/${file.filename}`;
    }

    const activityBeforeUpdate = await ActivityModel.findById(id);
    if (!activityBeforeUpdate) {
      return res.status(404).send({ error: "Activity not found" });
    }

    await ActivityModel.update(formData, {
      where: { ID_activity: id }
    });

    const activityAfterUpdate = await ActivityModel.findById(id);

    return res.send({ data: activityAfterUpdate });
  } catch (error) {
    handleHttpError(res, 'ERROR_UPDATE_ACTIVITY');
  }
}


```

Imagen 35 Controller PUT

Este controlador funciona de la misma manera que el anterior, debido a que las solicitudes provienen de una solicitud *Multipart/Form-Data*, primero se busca la actividad a actualizar, si se encuentra la actividad, se envían los campos que el usuario ha actualizado (incluida la foto en este caso) y se devuelve una respuesta de confirmación con la actividad creada.

```
/**  
 * Método que elimina una actividad dado su identificador.  
 * @param {*} req  
 * @param {*} res  
 * @returns  
 */  
const deleteActivity = async (req, res) => {  
    try {  
        const { id } = req.params;  
        const activity = await ScheduleModel.findOne({  
            where: { ID_Activity: id }  
        });  
        if (activity) {  
            return res.status(404).json({ error: "ACTIVITY_IN_SCHEDULE" });  
        }  
        const user = await ActivityModel.destroy({  
            where: { ID_activity: id }  
        });  
        if (user === 1) {  
            return res.json({ message: "Activity deleted successfully" });  
        } else {  
            return res.status(404).json({ error: "Activity not found" });  
        }  
    } catch (error) {  
        return handleHttpError(res, "ERROR_DELETE_ACTIVITY");  
    }  
};
```

Imagen 36 Controller DELETE

Este controlador refleja el borrado de una actividad, de la misma manera, se recibe como parámetro el identificador único de la actividad. En este caso se hace un tratamiento extra a la solicitud, ya que no se permite al usuario eliminar una actividad que haya sido asignada a un horario previamente.

Por lo tanto, primero se busca si existe algún horario cuya actividad sea la que se busca borrar, si esto ocurre se envía al usuario un mensaje informativo y no se permite borrar.

Finalmente, si no existe ningún horario cuya actividad sea la que se desea borrar, la actividad se borrará sin problemas y se enviará un mensaje informativo al usuario.

#### 5.1.9. Carpeta Controllers.

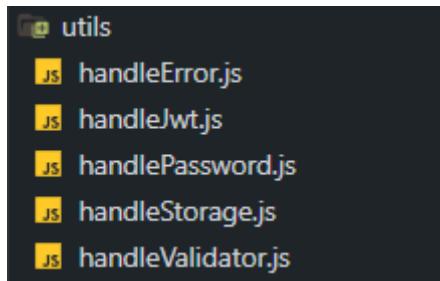
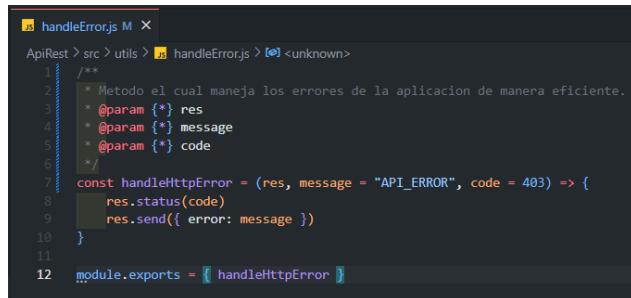


Imagen 37 Carpeta Utils

En esta carpeta se encuentran todas aquellas funciones recurrentes en la aplicación, son funciones cuya utilidad es a nivel global en el proyecto, y de esa manera se vuelve conveniente tener una capa donde poder organizar todos estos útiles.

En el próximo apartado se explican los distintos útiles que se han utilizado en la aplicación.



```

js handleError.js M ×
ApiRest > src > utils > handleError.js > [e] <unknown>
1 /**
2  * Metodo el cual maneja los errores de la aplicacion de manera eficiente.
3  * @param {*} res
4  * @param {*} message
5  * @param {*} code
6  */
7 const handleHttpError = (res, message = "API_ERROR", code = 403) => {
8   res.status(code)
9   res.send({ error: message })
10 }
11
12 module.exports = { handleHttpError }

```

Imagen 38 Error Útil

En este útil podemos gestionar los errores de manera más eficiente, se utiliza enviando la respuesta, un mensaje el cual se quiere plasmar al usuario y un código de respuesta (por defecto el 403 *Unauthorized*). De esta manera el control de errores se realiza en esta capa y no se envía información innecesaria al usuario.



```

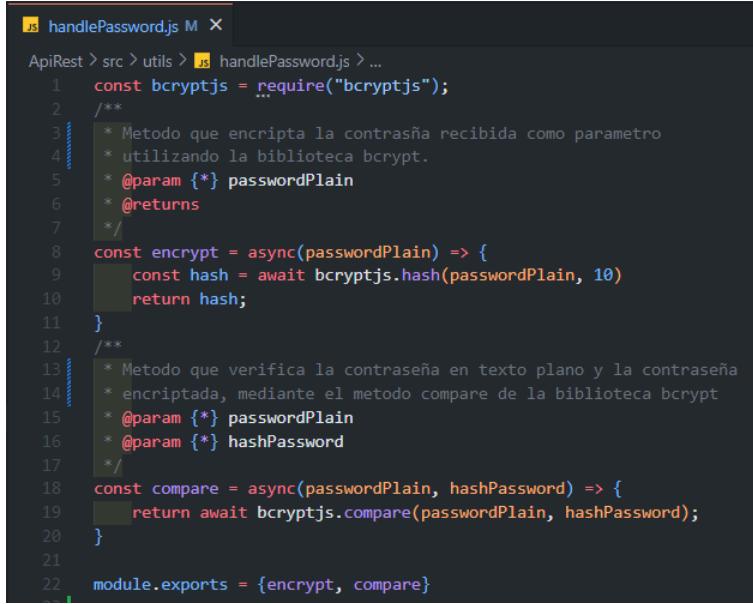
js handleJwt.js M ×
ApiRest > src > utils > handleJwt.js > ...
1 const jwt = require("jsonwebtoken");
2 const JWT_SECRET = process.env.JWT_SECRET;
3
4 /**
5  * Metodo el cual genera un Token con duracionde 2 horas
6  * @param {*} user
7  */
8 const tokenSign = async (user) => {
9   const sign = jwt.sign(
10     {
11       ID_user: user.ID_user,
12       Rol: user.Rol
13     },
14     JWT_SECRET,
15     {
16       expiresIn: "2h"
17     }
18   )
19   return sign;
20 };
21
22 /**
23  * metodo que recibe un token y verifica la validez del mismo
24  * @param {*} tokenJwt
25  * @returns
26  */
27 const verifyToken = async (tokenJwt) => {
28   try {
29     return jwt.verify(tokenJwt, JWT_SECRET)
30   } catch(e){
31     return null;
32   }
33 }
34
35 module.exports = { verifyToken, tokenSign }

```

Imagen 39 Token Útil

Este útil tiene dos métodos, en el primero se hace uso de la librería *JWT*, está librería nos permite generar tokens mediante una clave, estos tokens son únicos y solo reconocibles por el mismo método de encriptación. De esta manera se genera un token para el usuario para que pueda mantener una sesión y hacer las peticiones que le esté permitido hacer.

El segundo método se encarga la validez del token que recibe como parámetro por parte de la solicitud del usuario.

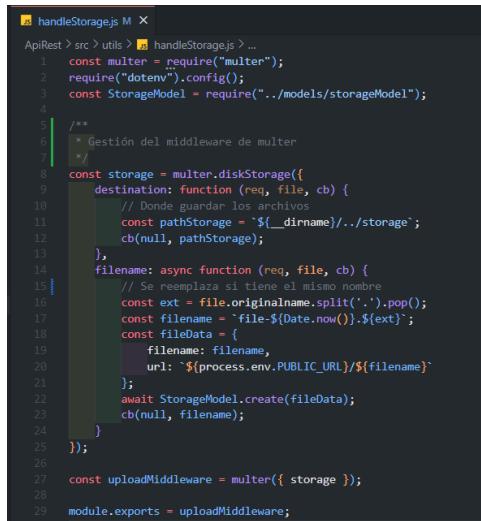


```
js handlePassword.js M X
ApiRest > src > utils > js handlePassword.js > ...
1  const bcryptjs = require("bcryptjs");
2  /**
3   * Metodo que encripta la contraseña recibida como parametro
4   * utilizando la biblioteca bcrypt.
5   * @param {*} passwordPlain
6   * @returns
7   */
8  const encrypt = async(passwordPlain) => {
9    const hash = await bcryptjs.hash(passwordPlain, 10)
10   return hash;
11 }
12 /**
13  * Metodo que verifica la contraseña en texto plano y la contraseña
14  * encriptada, mediante el metodo compare de la biblioteca bcrypt
15  * @param {*} passwordPlain
16  * @param {*} hashPassword
17 */
18 const compare = async(passwordPlain, hashPassword) => {
19   return await bcryptjs.compare(passwordPlain, hashPassword);
20 }
21
22 module.exports = {encrypt, compare}
```

Imagen 40 Encrypt Útil

Este útil gestiona todo lo que tiene que ver con la encriptación de la contraseña del usuario, todo esto mediante el uso de la biblioteca BCRYPT (Niels Provos & David Mazières, 1999). En el primer método se encripta una contraseña recibida como parámetro y se devuelve dicha contraseña para insertar en base de datos. De esta manera protegemos la información del usuario.

El segundo método se utiliza para comparar la validez de la contraseña, funciona de manera que, utilizando el método *compare* de la biblioteca *bcrypt*, se compara la contraseña en texto plano y su versión encriptada, esto devuelve una respuesta afirmativa si las contraseñas coinciden o manda una respuesta negativa si las contraseñas no coinciden.



```

1  const multer = require("multer");
2  require("dotenv").config();
3  const StorageModel = require("../models/storageModel");
4
5  /**
6   * Gestión del middleware de multer
7   */
8  const storage = multer.diskStorage({
9    destination: function (req, file, cb) {
10      // Donde guardar los archivos
11      const pathStorage = `${__dirname}/../storage`;
12      cb(null, pathStorage);
13    },
14    filename: async function (req, file, cb) {
15      // Se reemplaza si tiene el mismo nombre
16      const ext = file.originalname.split('.').pop();
17      const filename = `file-${Date.now()}.${ext}`;
18      const fileData = {
19        filename,
20        url: `${process.env.PUBLIC_URL}/${filename}`
21      };
22      await StorageModel.create(fileData);
23      cb(null, filename);
24    }
25  });
26  const uploadMiddleware = multer({ storage });
27
28 module.exports = uploadMiddleware;
  
```

*Imagen 41 Storage Util*

En este archivo se gestiona la subida de archivos a la base de datos, mediante la biblioteca *Multer*.

Se configura *Multer* de manera que, primero se establece la dirección de los archivos donde se van a almacenar físicamente, en este caso en la carpeta *Storage* del mismo proyecto.

Finalmente se crea un registro en la tabla *Almacenamiento*, para poder así asociar la imagen al usuario o actividad donde ha sido insertada.

De esta manera, mediante esta configuración, se permite a los usuarios hacer peticiones que conlleven archivos, cabe destacar que solo se permitirá el almacenamiento de archivos cuyo formato sea imagen, no se admitirá otro tipo de formato.



```

1  const { validationResult } = require("express-validator");
2
3  /**
4   * Método que valida la entrada de datos por parte del cliente.
5   * @param {*} req
6   * @param {*} res
7   * @param {*} next
8   * @returns
9  */
10 const validateResults = (req, res, next) => {
11   try {
12     validationResult(req).throw();
13     return next();
14   } catch (err) {
15     res.status(403);
16     res.send({ errors: err.array() });
17   }
18 }
19
20 module.exports = validateResults;
  
```

*Imagen 42 Validator Util*

Como se ha mencionado antes, se ha creado un útil el cual nos permite verificar la integridad de los datos que provienen del cliente, de esta manera evitando posibles solicitudes maliciosas.

Este útil se encarga de validar los datos de entrada a la API, mediante una serie de validadores, los cuales se explican a continuación.

### 5.1.10. Carpeta Validadores.

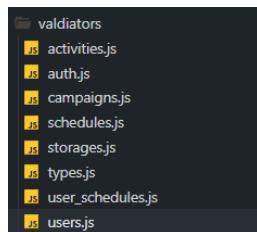
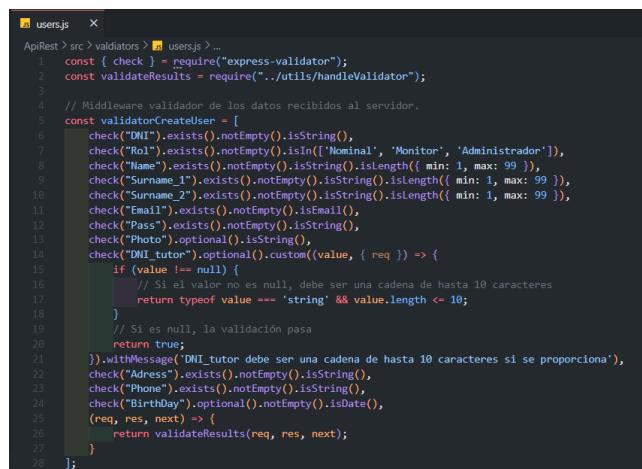


Imagen 43 Carpeta Validators

En esta carpeta se encuentran todos los validadores de información, de esta manera agregando seguridad al contacto que tiene la API con el cliente. Dentro de cada uno de estos archivos, se definen unas pautas con las cuales los datos de entrada deberán cumplir, si no cumplen con estas pautas, la solicitud será denegada.



```

1 const { check } = require("express-validator");
2 const validateResults = require("../utils/handleValidator");
3
4 // Middleware validador de los datos recibidos al servidor.
5 const validatorCreateUser = [
6   check("DNI").exists().notEmpty().isString(),
7   check("Rol").exists().notEmpty().isIn(["Nominal", "Monitor", "Administrador"]),
8   check("Name").exists().notEmpty().isString().isLength({ min: 1, max: 99 }),
9   check("Surname_1").exists().notEmpty().isString().isLength({ min: 1, max: 99 }),
10  check("Surname_2").exists().notEmpty().isString().isLength({ min: 1, max: 99 }),
11  check("Email").exists().notEmpty().isEmail(),
12  check("Pass").exists().notEmpty().isString(),
13  check("Photo").optional().isString(),
14  check("DNI_tutor").optional().custom((value, { req }) => {
15    if (value === null) {
16      // Si el valor no es null, debe ser una cadena de hasta 10 caracteres
17      return typeof value === 'string' && value.length <= 10;
18    }
19    // Si es null, la validación pasa
20    return true;
21  }).withMessage('DNI_tutor debe ser una cadena de hasta 10 caracteres si se proporciona'),
22  check("Address").exists().notEmpty().isString(),
23  check("Phone").exists().notEmpty().isString(),
24  check("Birthday").optional().notEmpty().isDate(),
25  (req, res, next) => {
26    return validateResults(req, res, next);
27  }
28];

```

Imagen 44 Validador User

Como se puede apreciar en el archivo, se declara un objeto, el cual contiene las restricciones para cada campo de la solicitud, para este caso comprueba que el campo de DNI tenga sea de tipo *string* y que no venga vacío el dato, comprueba que el rol mediante el cual se está registrando el usuario coincida con “*Nominal*”, “*Monitor*” o “*Administrador*”.

Hace una serie de comprobaciones al cuerpo de la solicitud, y requiere que todas sean correctas para poder validar la solicitud. De esta manera, agregando una capa más de restricción de acceso a datos, mejorando la calidad de los datos introducidos en la base de datos.

### 5.1.11. Estructura.

En conclusión, se ha diseñado una API tratando de cumplir con las normativas REST, de esta manera asegurando la protección de la misma y la escalabilidad, tanto vertical como horizontal.

Se ha tenido en cuenta un desarrollo modular, para que adopte una forma fácil de entender, ya sea para consultar o para escalar. Todas las pruebas de funcionamiento de la API se han hecho mediante *POSTMAN*, lo cual se hablará más adelante, así como la documentación de la misma.

## 5.2. Diseño del cliente.

Se ha utilizado el *framework* de Angular, que es un *framework* de desarrollo web, desarrollado y mantenido por Google, utilizado para construir aplicaciones web de una sola página (SPA), de manera eficiente y escalable. Como se ha mencionado anteriormente, se tomó la decisión de cambiar del *framework* de *React* a *Angular* y es por estas razones:

Angular proporciona una estructura clara y modular, facilitando la organización del código, además incluye herramientas y bibliotecas que aceleran el desarrollo, como la CLI de Angular.

También fomenta el uso de componentes reutilizables, mejorando la mantenibilidad y consistencia de la aplicación. Por lo tanto, ofrece un enlace bidireccional, para facilitar la comunicación entre sus componentes.

Finalmente, cuenta con un fuerte soporte de Google y una comunidad activa, lo que garantiza actualizaciones regulares con una amplia gama de recursos y soluciones disponibles.

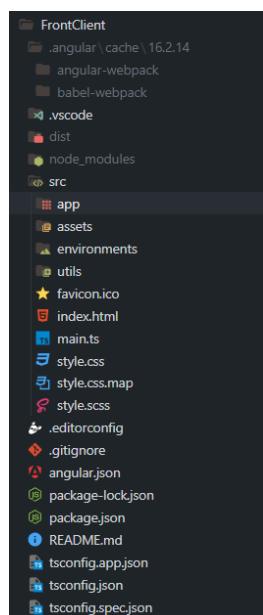
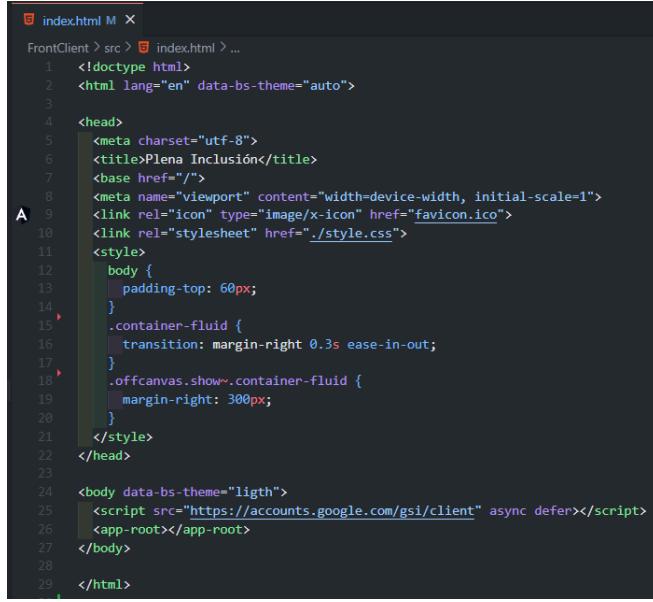


Imagen 45 Estructura Angular

Angular fomenta el orden dentro de sus proyectos, ofreciéndonos una estructura de carpetas bastante ordenada e intuitiva de base, base de la cual se partirá para poder desarrollar el proyecto de la manera en la que se puede ver en la imagen.

Al tratarse de un *framework* de aplicaciones de una sola página (SPA), contiene su archivo principal, el cual es la plantilla *html* donde se van a ir intercambiando todos los componentes de la aplicación.



```

index.html M X
FrontClient > src > index.html > ...
1  <!DOCTYPE html>
2  <html lang="en" data-bs-theme="auto">
3
4  <head>
5    <meta charset="utf-8">
6    <title>Plena Inclusión</title>
7    <base href="/">
8    <meta name="viewport" content="width=device-width, initial-scale=1">
9    <link rel="icon" type="image/x-icon" href="favicon.ico">
10   <link rel="stylesheet" href="./style.css">
11   <style>
12     body {
13       padding-top: 60px;
14     }
15     .container-fluid {
16       transition: margin-right 0.3s ease-in-out;
17     }
18     .offcanvas.show ~ .container-fluid {
19       margin-right: 300px;
20     }
21   </style>
22 </head>
23
24 <body data-bs-theme="light">
25   <script src="https://accounts.google.com/gsi/client" async defer></script>
26   <app-root></app-root>
27 </body>
28
29 </html>

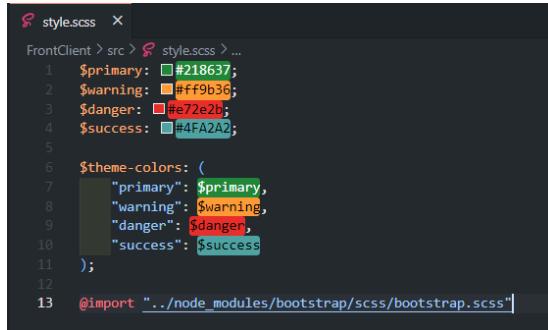
```

*Imagen 46 index Plena Iclusion*

Además, en el desarrollo del proyecto se ha hecho uso de Bootstrap, ya que nos permite tener un control bastante intuitivo en el aspecto del diseño de la página web. Claramente los colores que permite usar Bootstrap no se parecen a los colores corporativos que se necesita para llevar a cabo el diseño de la interfaz de la aplicación.

Debido a esto se ha decidido cambiar las clases de colores dentro de Bootstrap, con el objetivo de alcanzar lo máximo posible los colores deseados para la implementación. Para lograr esto se necesita instalar Bootstrap en la aplicación.

Una vez instalado Bootstrap, mediante una hoja de estilos “.scss”, se modifican las variables que nos brinda Bootstrap de esta manera:



```

style.scss X
FrontClient > src > style.scss > ...
1  $primary: #218637;
2  $warning: #ff9b36;
3  $danger: #e72e2b;
4  $success: #4FAZAA;
5
6  $theme-colors: (
7    "primary": $primary,
8    "warning": $warning,
9    "danger": $danger,
10   "success": $success
11 );
12
13 @import "../node_modules/bootstrap/scss/bootstrap.scss";

```

*Imagen 47 Paleta de colores*

Esto hará que se genere una hoja de estilos con todas las clases modificadas con los colores que necesitamos aplicar en la aplicación, como se puede ver en la imagen, se han modificado los colores que hacen referencia a las clases: *primary*, *warning*, *danger*, *success*.



Dentro de la carpeta `src` se encuentra la arquitectura de proyecto, seguidamente se va a explicar cada capa del proyecto, ya que al igual que la estructura del servidor y con las ventajas que nos proporciona Angular, se ha tratado de llevar una estructura escalable para el tiempo.

### 5.2.1. Carpeta Components.

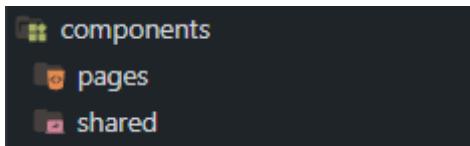


Imagen 48 Carpeta Components

En esta carpeta van situados todos los componentes de los cuales se va a hacer uso en la aplicación, estos componentes a la vez están divididos en dos carpetas: en la carpeta `pages` van todas las plantillas con las pantallas que se van a enseñar al usuario y en la carpeta `shared` van todos los componentes que van a conformar todas las plantillas de páginas.

A screenshot of a code editor showing the content of the file "register-page.component.html". The code is written in HTML and uses Bootstrap classes. It includes an `<app-form>` tag which is highlighted in green, indicating it's a component being used as a template.

Imagen 49 Componente de registro

Como se puede ver en el ejemplo, la página de registro simplemente sirve como plantilla y se trata de minimizar lo máximo posible la carga de trabajo de cada componente, haciendo que cada componente tenga una funcionalidad. La misma estructura siguen las demás componentes que representan las páginas donde el usuario va a poder navegar.

### 5.2.2. Carpeta Guards

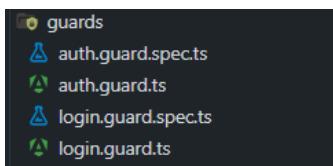


Imagen 50 Carpeta Guards

Una de las funcionalidades que nos proporciona Angular es la capacidad de proteger las rutas de acceso, y esto se logra mediante el uso de `guards`, es decir, cada ruta está monitorizada para que cuando el usuario intente acceder a ella, se verifique el rol del mismo y se le permita el acceso o se le redirija a otra página alternativa.



Como se puede notar en la imagen, para el desarrollo de este proyecto se han utilizado dos guards, uno se utiliza para comprobar que el usuario tiene una sesión activa, de lo contrario sería redirigido a la página de inicio de sesión.

El segundo *guard* se utiliza en el caso de: si un usuario tiene una sesión iniciada, no se le permite acceder a la pantalla de inicio de sesión, se le redirige a la página principal de la aplicación.

#### 5.2.3. Carpeta Interceptors

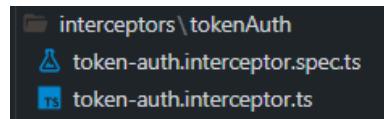


Imagen 51 Carpeta Interceptors

Otra de las funcionalidades que brinda Angular es la posibilidad de interceptar todas las peticiones que salen de sus servicios, esto se hace principalmente para editar las cabeceras de las peticiones de una manera sencilla y global, ya que configurar esto hará que todas las cabeceras de las solicitudes se vean modificadas.

En este caso se modifica enviando el token del usuario, de esta manera se podrá acceder a las rutas que estaban protegidas en nuestro servidor. Con este token se validará la identidad del usuario que inicia sesión y quiere navegar por la aplicación.

#### 5.2.4. Servicios.

Una buena práctica en Angular es hacer uso de los servicios, estos actúan como conexión con el servidor, permitiendo enviar y recibir dato de una manera simple, ya que estos mismos servicios pueden ser inyectados en cualquier componente en cualquier parte de la aplicación. Con esto se garantiza no duplicar código y hacer un buen uso de las herramientas que nos proporciona Angular.

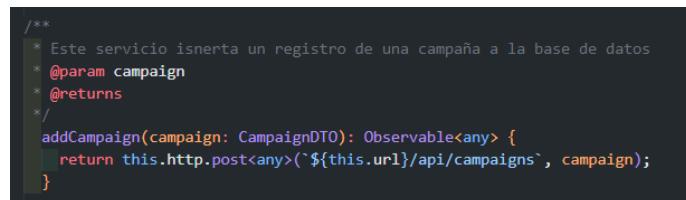


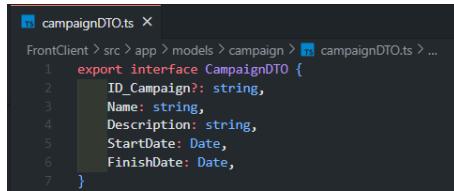
Imagen 52 Servicio en Angular

En este ejemplo se puede ver como se conforma un servicio, recibe como parámetro un objeto de transmisión de datos (DTO) y hace la solicitud al *endpoint* correspondiente del servidor, para luego recibir una respuesta en forma de *Observable*, el manejo de esa respuesta se hace en cada componente que hace uso de este servicio.

### 5.2.5. Carpeta Models

En esta carpeta se almacenan todos los Objetos de Transmisión de Datos (DTO), cada uno hace referencia a una tabla de la base de datos, es buena práctica realizar estos objetos, debido a que así se puede limpiar lo máximo posible la información que se recibe por parte del servidor.

En la siguiente imagen se puede ver como se genera un objeto, el cual contiene los datos necesarios que recibe por parte del servidor, luego estos datos serán interpretados por el componente correspondiente.



```

campaignDTO.ts
FrontClient > src > app > models > campaign > campaignDTO.ts > ...
1  export interface CampaignDTO {
2    ID_Campaign?: string,
3    Name: string,
4    Description: string,
5    StartDate: Date,
6    FinishDate: Date,
7  }
  
```

Imagen 53 Modelo Campaign

### 5.2.6. Carpeta Pipes

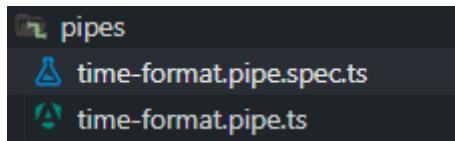


Imagen 54 Carpeta Pipes

A veces no solo basta con tratar de manera correcta los datos que nos devuelve el servidor mediante un DTO, sino que se necesita dar un formato a esos datos. Para ello Angular permite hacer uso de los *pipes*, esta es una herramienta con la cual podemos dar el formato deseado a los datos recibidos por parte del servidor, como por ejemplo: recibimos una fecha en formato HH:MM:SS y mediante el uso del *Pipe* podemos cambiar el formato a HH:MM.

### 5.2.7. Archivo Environments

Angular proporciona una manera sencilla de administrar toda la información sensible dentro de la aplicación, mediante el uso de *Environments*, dentro de este archivo se van a declarar todas las variables que se quieren utilizar dentro de la aplicación, de esta manera se pueden cargar datos sensibles que no convenga que se encuentren dentro de los componentes.



```

environments.ts
FrontClient > src > environments > environments.ts > ...
1  export const environments = {
2    baseUrl: 'http://localhost:3000',
3    defaultProfileImage: "1fd5abd7-0784-41b7-814e-0755560d50c8",
4    noActivityImage: "7b60d078-068f-4856-9515-510a43116d7a",
5    plenaInclusionLogo: "a835bda1-a96b-43e3-948f-cbb16ccca6ef5",
6    client_id: '562380291200-abf1rofq5kjldohfj45nho30bd17c.apps.googleusercontent.com',
7    plenaInclusionToast: '24831c35-b500-4751-9191-1ab19904d601'
8  }
  
```

Imagen 55 Archivo Environments

### 5.2.8. Carpeta Utils

Esta carpeta contiene las funciones que se reutilizan en todo el proyecto, llamados útiles debido a que son reutilizables por varios componentes en toda la aplicación.

Un ejemplo es la validación de datos para los formularios, se comprueba que no se inserten caracteres extraños y se devuelve una respuesta, positiva o negativa, según el carácter recibido como parámetro.

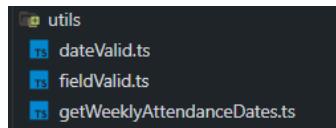


Imagen 56 Carpeta Utils

### 5.2.9. Estructura

La estructura de la parte de cliente se ha tratado que sea lo más intuitiva posible, tratando de modularizar lo máximo posible, cumpliendo con el principio de mínima funcionalidad, el cual dicta que un componente tiene que encargarse de una tarea a ser posible.

Se ha desarrollado el proyecto de manera que añadir funcionalidades sea sencillo y escalable, además cada apartado está debidamente aislado para cumplir con cada una de las funcionalidades del cliente, listo para poder agregar nuevas funcionalidades para el futuro.

## 5.3. Conexión con los servicios RDS de Amazon Web Service.

Para el almacenamiento de los datos, se ha optado por hacer uso del servicio RDS que ofrece Amazon Web Service. Este servicio permite a los usuarios manejar bases de datos como MySQL, PostgreSQL, MariaDB, Oracle y SQL Server, además ofrece características como copias de seguridad automáticas, recuperación ante desastres y escalado flexible, sin la necesidad de administrar una infraestructura subyacente.

Para hacer uso de este servicio se tiene que seguir varios pasos, empezando por la creación de la cuenta de Amazon Web Service.



Imagen 57 Servicio RDS

### 5.3.1. Pasos para implementar el servicio RDS de Amazon Web Service.

#### Crear cuenta en Amazon Web Service:

Para poder hacer uso de todos los servicios que ofrece esta plataforma, es necesario un registro previo. Para ello se deberá llenar varios formularios donde se piden información necesaria para este procedimiento, cabe señalar que esta plataforma ofrece un perfil con costes mínimos e incluso nulos para aquellas personas cuyo fin sea educativo.

#### Seleccionar el servicio RDS dentro de las opciones Bases De Datos:

Amazon Web Service cuenta con una extensa variedad de servicios para ofrecer a sus clientes, en este caso se hará uso del servicio RDS y se creará la base de datos.



Imagen 58 Servicio RDS BBDD

#### Seleccionar el motor de la base de datos:

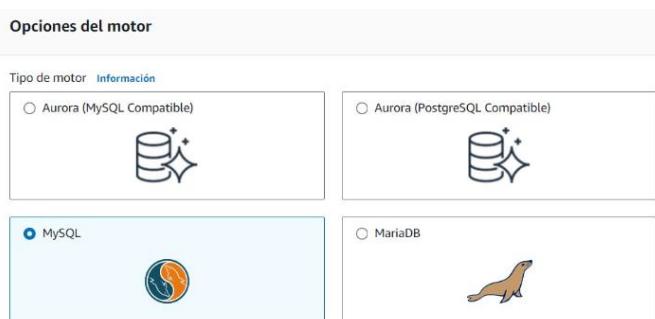


Imagen 59 Motor de base de datos

Dentro de este apartado se configuran los aspectos importantes a la hora de la creación de la base de datos, como puede ser el motor con el que se va a crear. En este caso MySQL.



## Elegir la platilla:

The screenshot shows a "Plantillas" (Templates) section with three options:

- Producción**: Utilice los valores predeterminados para disfrutar de una alta disponibilidad y de un rendimiento rápido y constante.
- Desarrollo y pruebas**: Esta instancia se ha diseñado para su uso en desarrollo, fuera de un entorno de producción.
- Capa gratuita**: Utilice el nivel gratuito de RDS para desarrollar nuevas aplicaciones, probar aplicaciones existentes o adquirir experiencia práctica con Amazon RDS.  
[Información](#)

*Imagen 60 Plantilla de base de datos*

En este caso se hace uso de la plantilla gratuita, esta plantilla permite a los usuarios desarrollar nuevas aplicaciones, probar existentes o adquirir experiencia. Lo que se ajusta perfectamente a las necesidades del proyecto.

## Habilitar el acceso público:

The screenshot shows the "Acceso público" (Public Access) section with two options:

- Si**: RDS asigna una dirección IP pública a la base de datos. Las instancias de Amazon EC2 y otros recursos fuera de la VPC pueden conectarse a la base de datos. Los recursos de la VPC también pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.
- No**: RDS no asigna una dirección IP pública a la base de datos. Solo las instancias de Amazon EC2 y otros recursos dentro de la VPC pueden conectarse a la base de datos. Elija uno o varios grupos de seguridad de VPC que especifiquen qué recursos pueden conectarse a la base de datos.

*Imagen 61 Habilitar Acceso Publico*

Es necesario habilitar este apartado, debido a que se va a conectar a este servicio mediante nuestro servidor, en caso contrario AWS restringe todo tipo de conexiones.

The screenshot shows the "Estimated Monthly costs" section with the following breakdown:

	Cost
Instancia de base de datos	12.41 USD
Almacenamiento	2.40 USD
Total	14.81 USD

Esta estimación de facturación se basa en el uso bajo demanda, tal como se describe en [Precios de Amazon RDS](#). La estimación no incluye los costos de almacenamiento de copias de seguridad, operaciones de E/S (si proceden) ni transferencia de datos.

Realice una estimación de sus costos mensuales de la instancia de base de datos mediante la [Calculadora costo mensual AWS](#).

*Imagen 62 Costes RDS*

Una vez configurada la estructura de la base de datos, Amazon Web Service hará un cálculo de los costes de uso del servicio. Esto es solo informativo ya que se ha elegido la capa gratuita del servicio, y superar el uso que brinda esta capa es casi imposible.



Bases de datos (1)

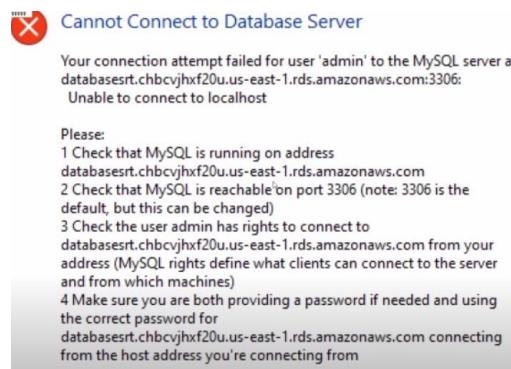
Recursos del grupo Modificar Acciones Restaurar desde S3 Crear base de datos

Filtrar por bases de datos

Identificador de base de datos	Estado	Role	Motor	Región y AZ	Tamaño	Recomendaciones
plenaInclusionbdd	Disponible	Instancia	MySQL Community	eu-north-1b	db.t3.micro	

*Imagen 63 Servicio RDS Creado*

Una vez creado el servicio y creado el gestor de base de datos, se puede hacer conexión al mismo. Para ello se necesita WorkBench y se necesita crear una nueva conexión con las credenciales que brinda el servicio. Cabe destacar que para realizar esta conexión se debe configurar reglas de entrada y salida, sino aparecerá el siguiente error.

*Imagen 64 Error de Conexión*

Para solucionar este error, como se ha mencionado antes, se deben añadir reglas de entrada y de salida de información dentro del servicio, y además, estas mismas reglas de entrada y de salida tienen que ir asignadas a la creación de la base de datos creada anteriormente.

Reglas de entrada

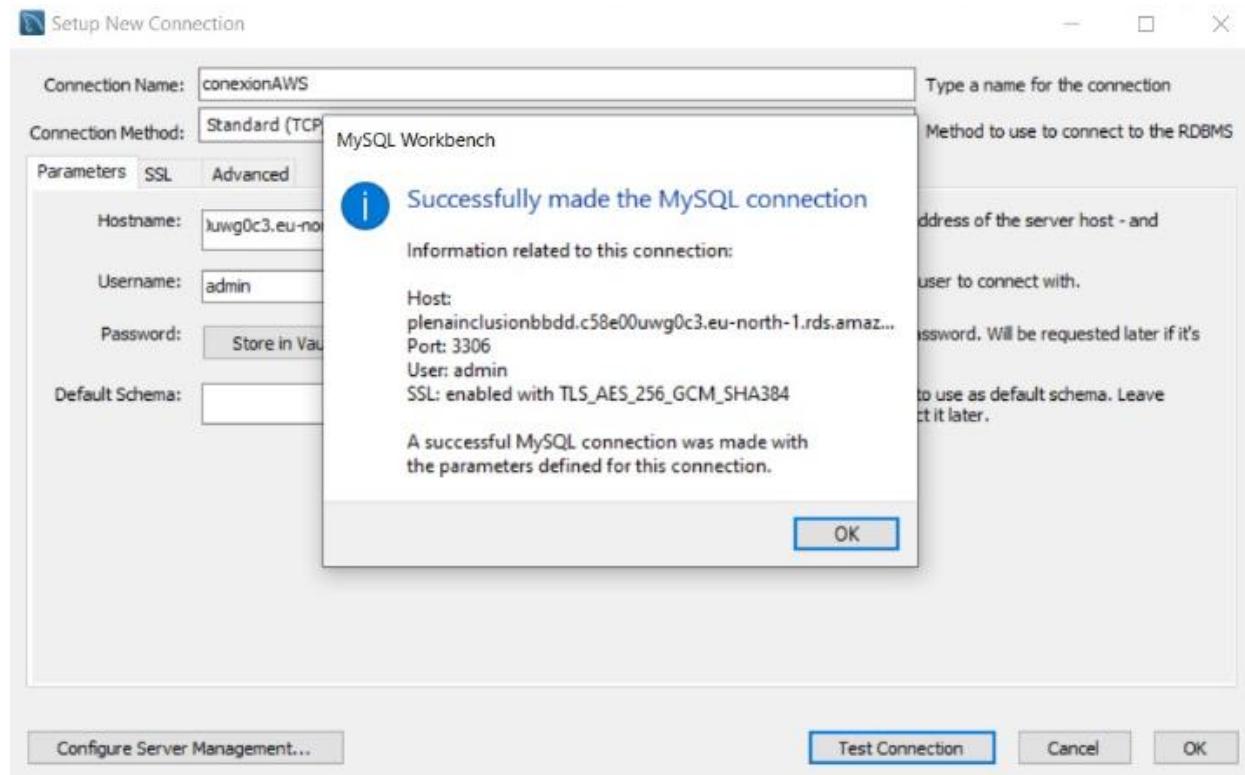
Tipo	Protocolo	Intervalo de puertos	Origen	Descripción: opcional
Todo el tráfico	Todo	Todo	Anywh...	0.0.0.0/0

Agregar regla Eliminar

*Imagen 65 Reglas de Entrada*



Una vez realizados todos estos pasos, se podrá hacer una conexión con el servicio previamente creado, de esta manera se podrá almacenar los datos de la aplicación en la nube. Permitiendo el acceso a los mismos desde cualquier aplicación.



*Imagen 66 RDS en Funcionamiento*

De esta manera todos los datos que se introduzcan en la aplicación estarán alojados en la nube.

## 6. Pruebas de funcionamiento y resultados obtenidos.

### 6.1. Tabla de verificación de requisitos funcionales.

Requisitos Funcionales	Completado/No Completado
La página tiene que tener un uso destinado para un usuario y un uso destinado para un profesional.	Completado
Los profesionales se registrarán de manera manual mientras que los usuarios necesitarán de un profesional para su registro.	Completado
Inicio de sesión de manera tradicional y mediante Google	Completado
Registro precio para acceder a todas las funcionalidades de la aplicación.	Completado
Crear actividades para los usuarios	Completado
Todo tiene que ir en lectura fácil.	Completado
Notificar al usuario mediante correo sobre las actividades	Completado
Lista de espera para las actividades	Completado
Formularios para calificar la actividad	Completado
Crear distintos programas para las actividades durante el año.	Completado
Los profesionales podrán ver una lista con todos los usuarios registrados.	Completado
Diseño corporativo.	Completado
Vista para el usuario amigable e intuitiva	Completado
Implementación de Google Calendar	No Completado

## 6.2. Pruebas realizadas.

Todas las pruebas de funcionamiento se encuentran detalladas en el anexo referencia a Anexo 2 *Pruebas de Funcionamiento*.

Cabe destacar que se han realizado las pruebas necesarias para ajustarse a los requisitos del proyecto por parte del cliente.

# 7. Conclusiones.

## 7.1. Resultados obtenidos.

Para ver los resultados obtenidos, consultar el anexo del manual de usuario., referencia a Anexo 3 *Manual de Usuario*.

## 7.2. Ampliaciones y mejoras.

A continuación, se describen las ampliaciones y mejoras de la aplicación.

### 7.2.1. Enviar Errores a Slack.

En el mundo del back-end es necesario tener un sistema que avise al desarrollador cuando está pasando un error, un sistema de monitorización de errores o de alertas. Para ello existen múltiples herramientas, unas mejores que otras.

*Slack* (Slack Technologies, 2013) es una plataforma de colaboración y comunicación diseñada para equipos. Permite la comunicación a través de canales organizados por temas, proyectos o equipos específicos, además de ofrecer mensajería directa o incluso videollamadas.

Mediante un middleware podemos integrar un bot que intercepte todas las solicitudes que recibe el servidor, de esta manera, se podrá ver información de quien ha hecho la petición, y a qué end-point ha hecho dicha solicitud, información que puede ser útil.

Pero también se puede interceptar los errores que emite el proceso del end-point en cuestión, y mediante la plataforma *Slack*, enviar dicho mensaje al canal. Todo esto para poder llevar un control a tiempo real de todos los errores que ocurren en la aplicación.



Imagen 67 Slack

### 7.2.2. Servicio EC2 de Amazon Web Service

EC2 (Elastic Compute Cloud) de Amazon Web Service es un servicio de computación en la nube que permite desplegar y gestionar instancias virtuales, proporcionando una estructura escalable y flexible para alojar aplicaciones web. De esta manera se podría subir la aplicación desarrollada a la web, para que pueda ser accesible desde cualquier dispositivo.

### 7.2.3. Incorporación de Tailwind y ShadCN.

En este proyecto se ha llevado el tema de los estilos mediante Bootstrap que, aunque sea muy eficaz, fácil y sencillo de implementar, es una herramienta que se está quedando en el pasado. Por ello una mejora visual al proyecto sería la incorporación de Tailwind (Adam Wathan & Steve Schoger, 2017) y ShadCN (*Shadcn/UI*, n.d.).

Tailwind es una framework de CSS de utilidad que proporciona clases predefinidas para aplicar estilos, estas clases cubren una amplia gama de propiedades permitiendo a los desarrolladores construir diseños personalizados sin escribir CSS personalizado. Tailwind es altamente configurable y facilita la creación de interfaces de usuario.



Imagen 68 Tailwindcss

ShadCN es una biblioteca de componentes accesibles y estilizados que se pueden usar junto con Tailwind para mejorar la consistencia y accesibilidad de un proyecto.

De esta manera se conseguiría una implementación de un software que despunta más en estos tiempos y que ofrece una cantidad de componentes que facilitan el desarrollo de la misma aplicación.



Imagen 69 ShadCN

### 7.3. Estimación del tiempo empleado.

Para llevar a cabo el desarrollo de este proyecto, se ha optado por seguir la metodología Agile (*Metodología Agile: Qué Es y Cómo Aplicarla a Tu Proyecto*, 2001), con un enfoque particular en Scrum.

Utilizando *Spins* de una semana se ha logrado iterar rápidamente e integrar funcionalidades de forma continua. En cada sprint se realiza una planificación del trabajo a realizar en la semana siguiente y se revisa el trabajo realizado en la semana previa.

En este caso se ha adoptado los roles de *Scrum Master*, *Product Owner* y equipo de desarrollo por una persona (el desarrollador), referencia a *Diagrama 6 Desarrollo Gantt*. Gracias a ello se ha podido cumplir con todos menos un requisito funcional. Referencia a *Diagrama 7 Gantt*



Se ha hecho uso de la herramienta de *Instagantt* para llevar un control del progreso, además, debido a que se ha utilizado un repositorio remoto mediante GitHub, se puede presentar una gráfica donde se representa el desarrollo del trabajo.

En la siguiente imagen se muestra un ejemplo de cómo se ha gestionado el uso de las ramas para el proyecto.

La rama principal “dev”, es la rama de la cual nacen las ramas con su tarea correspondiente, una vez finalizado el desarrollo de la tarea se realiza una *pull request* hacia la rama principal y se borra la rama anterior, de esta manera nunca se trabaja sobre la rama principal, sino desde ramas derivadas y luego el trabajo en estas ramas se juntan a la rama principal.

No han existido conflictos de ramas debido a que solo trabaja un desarrollador, así que no ha habido ningún impedimento en este modelo.

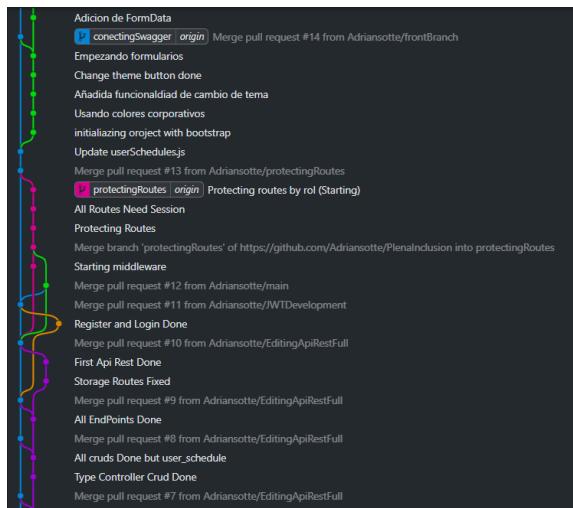


Imagen 70 Git Graph

#### 7.4. Valoración personal.

Para mi este proyecto ha sido bastante enriquecedor, me ha permitido aprender sobre estructuras de proyectos y nuevas formas de llevar a cabo tareas que no habíamos visto en clase, he aprendido sobre todo de cómo llevar una estructura más profesional de proyecto, de manera que pueda ser escalable y más legible por otros programadores o compañeros.

Además, he tenido la suerte de poder aprender sobre tecnologías que hoy en día están en auge, como puede ser *Angular* y *AWS*, esto me ha llevado a querer profundizar más sobre estas tecnologías e incluso me he encontrado con otras como puede ser *Vue* para desarrollo de *front* y *Azure* que es el gestor de la nube de *Microsoft*.

Creo que no he podido escoger mejor trabajo de fin de grado, ya que he podido poner en práctica todo lo aprendido en estos dos años, e incluso llevarlo a un nivel extra, teniendo en cuenta



claramente que tengo muchas cosas por aprender.

He podido aplicar mis conocimientos de base de datos, he podido aplicar mis conocimientos de JavaScript y lógica de código, he podido poner en práctica mis conocimientos de diseño de páginas web, también he podido aplicar mis conocimientos de arquitectura de un servidor para comunicarlo con un cliente.

Al final creo que he recopilado todo lo aprendido en un trabajo final, el cual estoy orgulloso de presentar, aunque me falta tiempo para todo lo que me gustaría haber agregado (gestión de errores, un diseño basado en componentes ya existentes).

También he ido cambiando varios aspectos de la aplicación según iba avanzando, ya que como en el trabajo desarrollo la misma tarea (desarrollo de páginas web), me he ido dando cuenta de cosas que he ido aprendiendo y aplicando en mi proyecto, lo cual no era tan eficiente porque cambiaba varias veces lo que ya tenía hecho, solo por dejarlo “más profesional”.

En definitiva, es un proyecto que está hecho con todo mi cariño y dedicación, me ha costado hacerlo, he tenido muchos problemas por el camino, pero el resultado creo que está acorde con el esfuerzo invertido.

Agradezco con todo el corazón a mis profesores, que son los que me han brindado los conocimientos que tengo hasta el día de hoy (no son fáciles de adquirir), y por haberme hecho conocer esta profesión que me hace brillar en lo que yo creo que es uno de mis puntos fuertes, la curiosidad y la dedicación.

Finalmente, espero que esta aplicación pueda servir de ayuda a la asociación a la que va dirigida, soy realista y sé que una aplicación web tiene que tener muchas más características de las que de momento no soy conocedor, y, por lo tanto, no he podido aplicar. En el transcurso del trabajo me iba dando cuenta de que hacer una página web totalmente funcional requiere de más personas, más tiempo y más conocimiento, pero estoy satisfecho con el trabajo que he desarrollado.

## 8. Bibliografía.

- Abhinav Asthana, A., & Sobti, A. K. (2012). *Postman API Platform | Sign Up for Free*. <https://www.postman.com/>
- Adam Wathan, & Steve Schoger. (2017). *Tailwind CSS - Rapidly build modern websites without ever leaving your HTML*. <https://tailwindcss.com/>
- Depold, S. (2012). *Getting Started | Sequelize*. <https://sequelize.org/docs/v6/getting-started/>
- Dustin Moskovitz, & Justin Rosenstein. (2008). *Instagantt*. <https://app.instagantt.com/r>
- Hevery, M. (2016, September 15). *Angular*. <https://angular.io/>
- Jordan Walke. (2013). *React*. <https://es.react.dev/>
- Larry Page, & Sergey Brin. (1998). *Company – Google*. <https://www.google.com/intl/en/about/company/>
- Lienhart Dahl, R. (2009, May 27). *Index | Node.js v21.7.1 Documentation*. <https://nodejs.org/docs/latest/api/>
- Mark Otto, & Jacob Thornton. (2011). *Get started with Bootstrap · Bootstrap v5.3*. <https://getbootstrap.com/docs/5.3/getting-started/introduction/>
- Matt Garman, & C.J. Moses. (2002). *AWS | Cloud Computing - Servicios de informática en la nube*. <https://aws.amazon.com/es/>
- Metodología Agile: qué es y cómo aplicarla a tu proyecto.* (2001). <https://blog.hubspot.es/marketing/metodologia-agile>
- Michael B. Jones, & John Bradley. (2010). *JSON Web Tokens - jwt.io*. <https://jwt.io/>
- Microsoft. (2015, April 29). *Documentation for Visual Studio Code*. <https://code.visualstudio.com/docs>
- Niels Provos, & David Mazières. (1999). *Bcrypt-Generator.com - Generate, Check, Hash, Decode Bcrypt Strings*. <https://bcrypt-generator.com/>
- Plena Inclusión Aragón. (2008). *Plena Inclusión Aragón*. <https://www.plenainclusionaragon.com/>
- Sequelize | Feature-rich ORM for modern TypeScript & JavaScript*. (n.d.). Retrieved June 10, 2024, from <https://sequelize.org/>
- shadcn/ui*. (n.d.). Retrieved June 10, 2024, from <https://ui.shadcn.com/>
- Slack Technologies. (2013). *Slack es tu plataforma de productividad | Slack*. <https://slack.com/intl/es-es>
- Subins2000. (2017, January 14). *draw.io*. <https://app.diagrams.net/>
- TJ Holowaychuk. (2012). *Express - Infraestructura de aplicaciones web Node.js*. <https://expressjs.com/es/>
- Tom-Preston-Werner, Wanstrath, C., Hyett, P. J., & Chacon, S. (2008, February 8). *GitHub*. <https://github.com/>
- Widenius, M., Axmark, D., & Larsson, A. (2001, January 1). *MySQL Documentation*. <https://dev.mysql.com/doc/>



Widenius, M., Larsson, A., & Axmark, D. (2005, September 1). *MySQL Workbench*.  
<https://www.mysql.com/products/workbench/>

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)



## 9. Anexo 1 Diagramas.

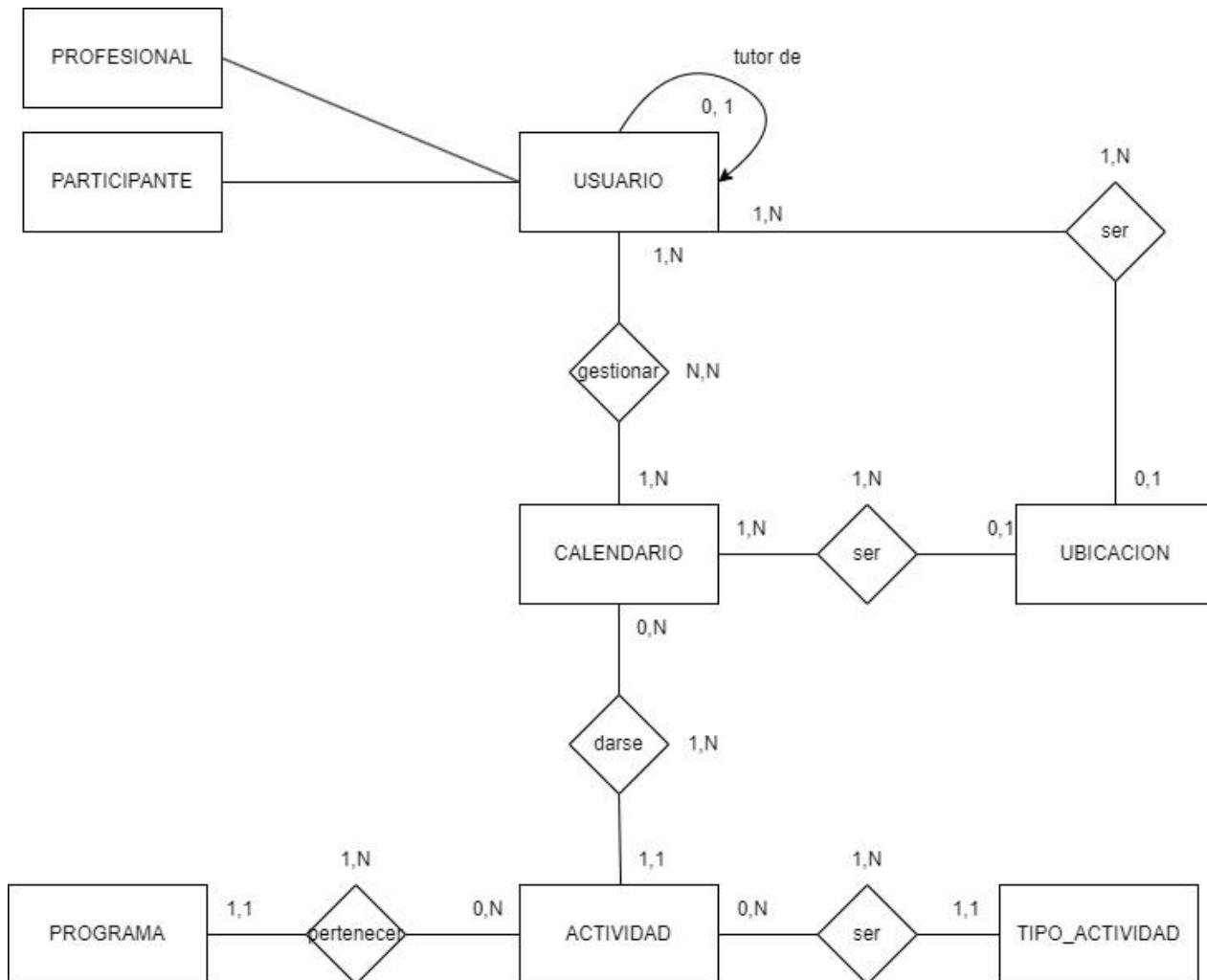
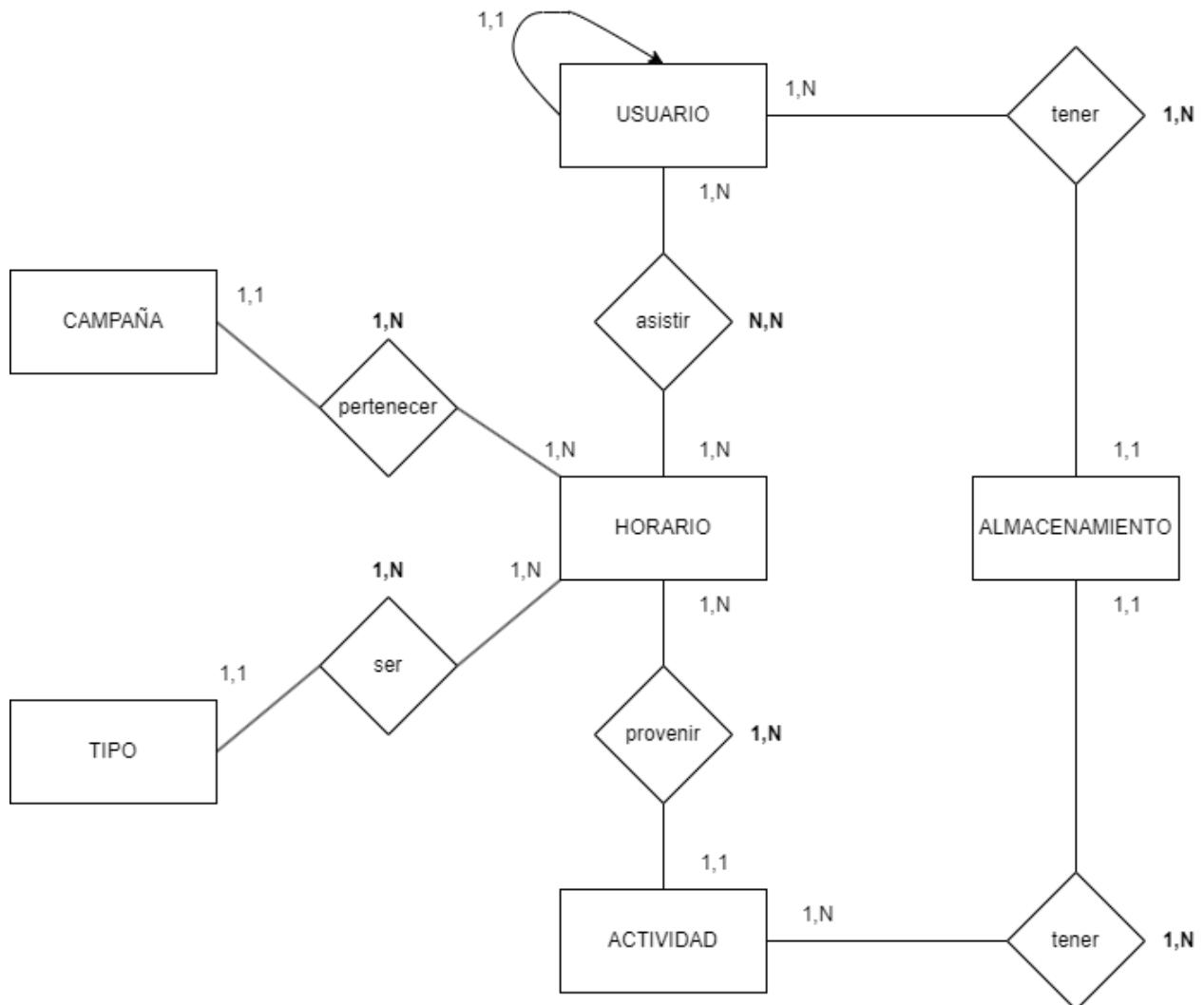


Diagrama 1 Entidad Relación Previo



### *Diagrama 2 Entidad Relación Actual*

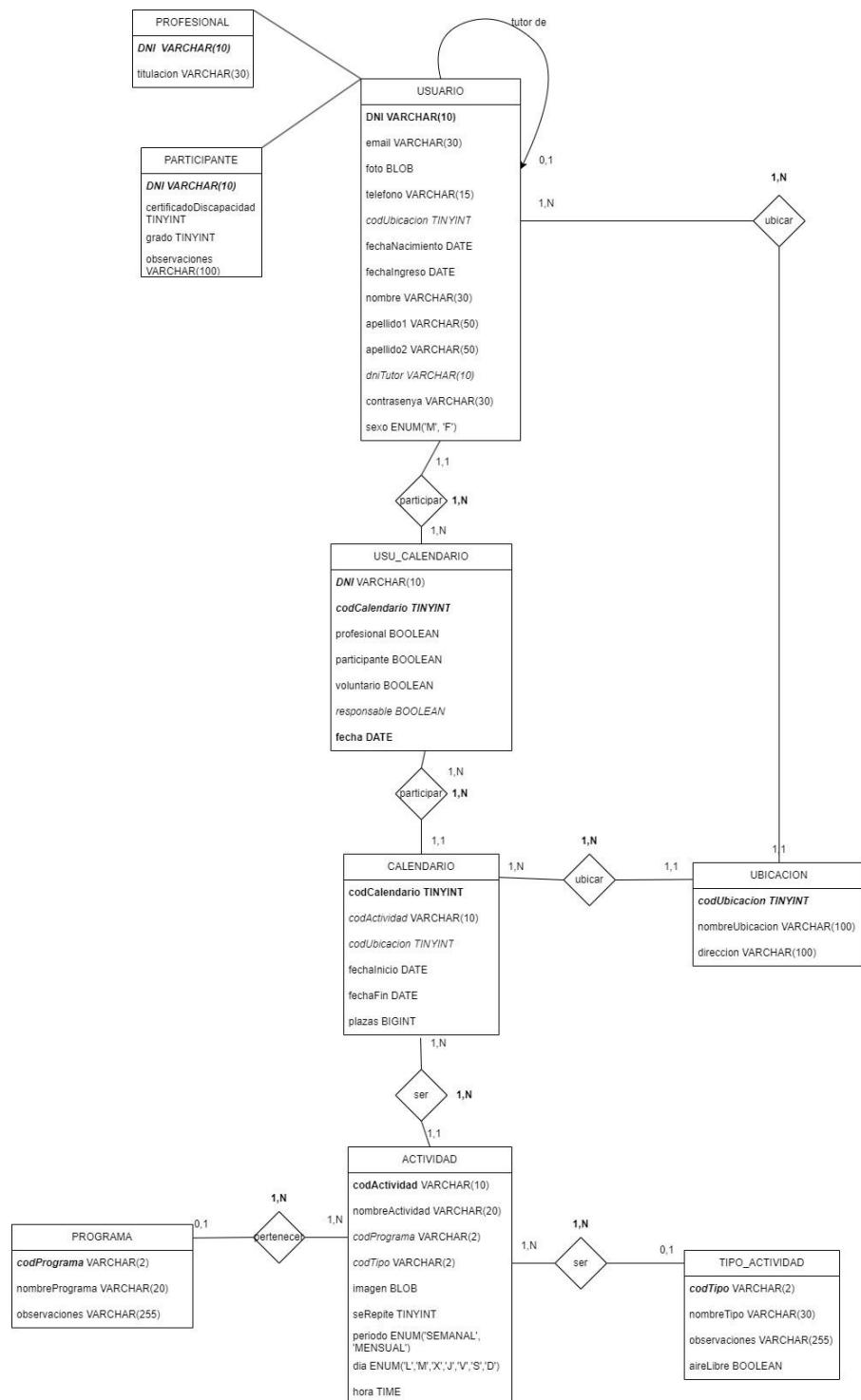


Diagrama 3 Relacionable Previo

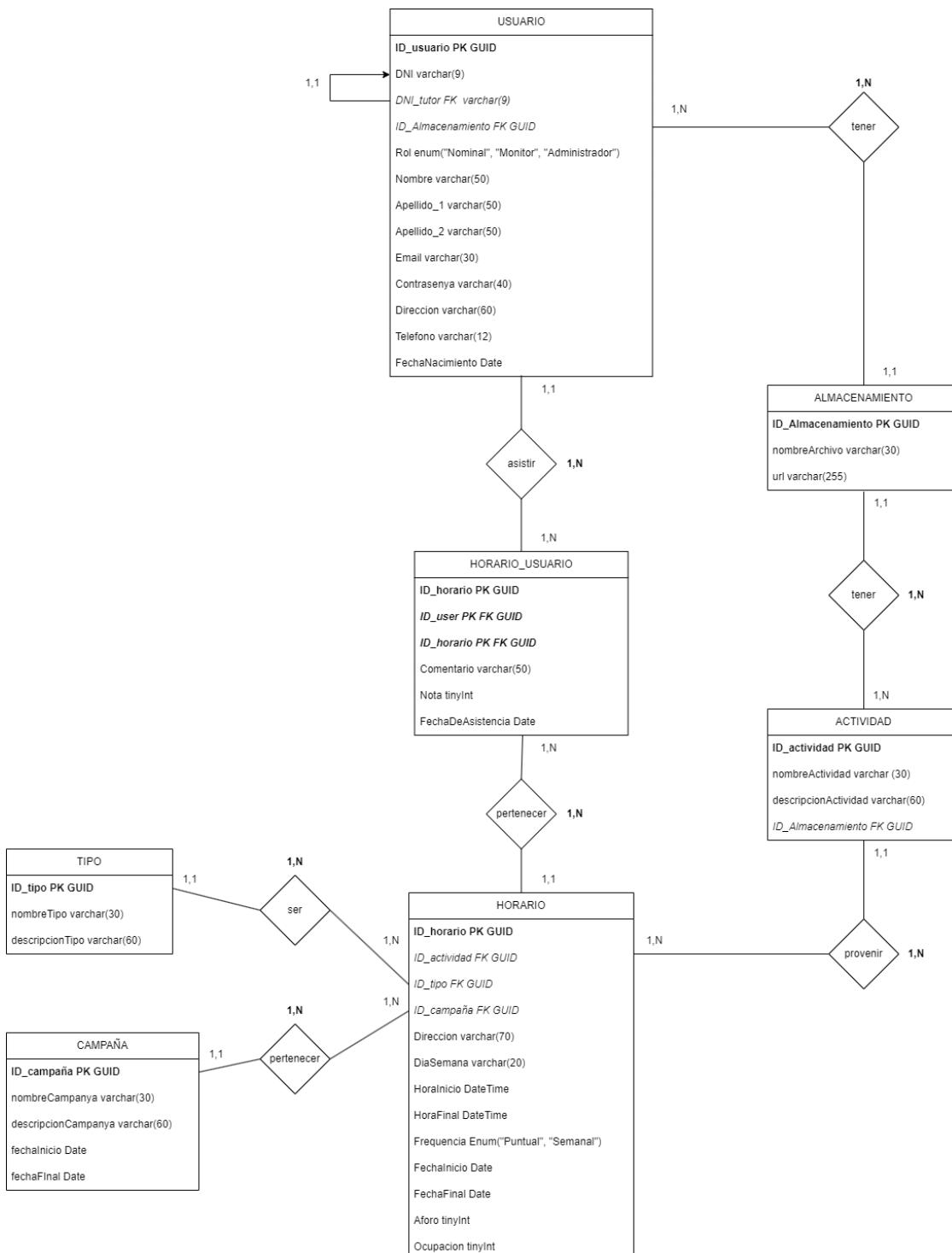


Diagrama 4 Relacionable Actual

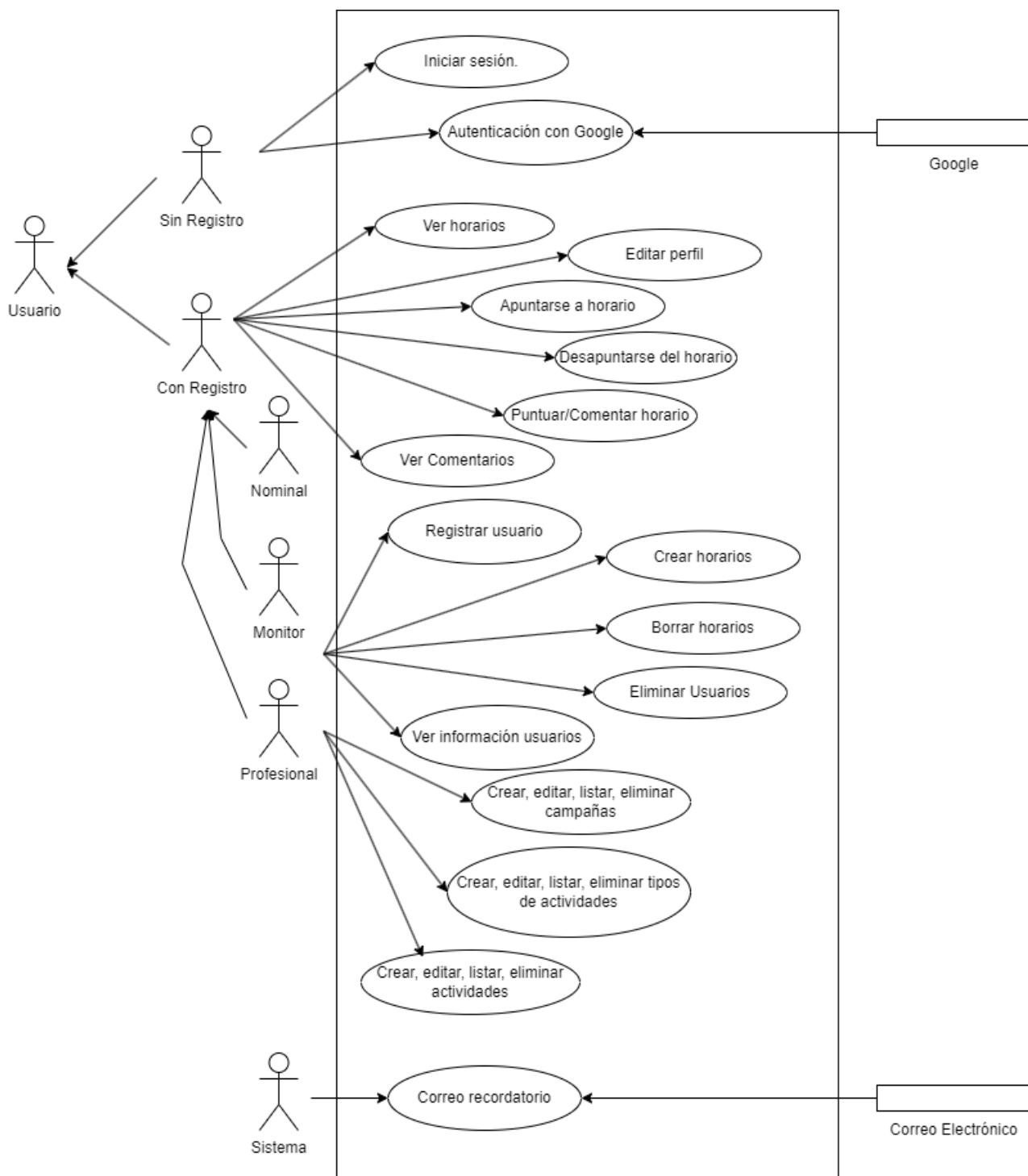
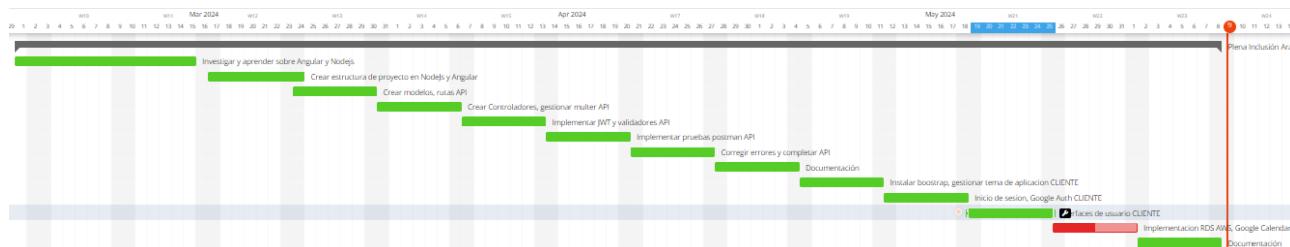


Diagrama 5 Caso de Usos

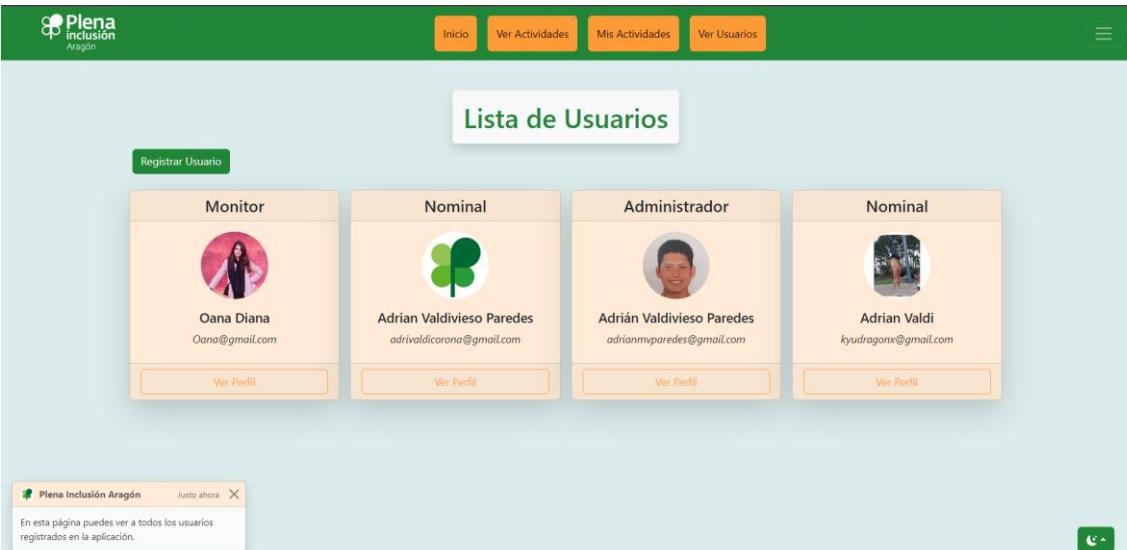
<b>Plena Inclusión Aragón Desarrollo:</b>			260h	01/Mar	08/Jun		96%
1	<input checked="" type="checkbox"/> Investigar y aprender sobre Angular y Nodejs	Adrián Valdivieso...	40h	01/Mar	15/Mar	Finished	100%
2	<input checked="" type="checkbox"/> Crear estructura de proyecto en Nodejs y Angular	Adrián Valdivieso...	-	17/Mar	24/Mar	Finished	100%
3	<input checked="" type="checkbox"/> Crear modelos, rutas API	Adrián Valdivieso...	10h	24/Mar	30/Mar	Finished	100%
4	<input checked="" type="checkbox"/> Crear Controladores, gestionar multer API	Adrián Valdivieso...	15h	31/Mar	06/Apr	Finished	100%
5	<input checked="" type="checkbox"/> Implementar JWT y validadores API	Adrián Valdivieso...	20h	07/Apr	13/Apr	Finished	100%
6	<input checked="" type="checkbox"/> Implementar pruebas postman API	Adrián Valdivieso...	10h	14/Apr	20/Apr	Finished	100%
7	<input checked="" type="checkbox"/> Corregir errores y completar API	Adrián Valdivieso...	15h	21/Apr	27/Apr	Finished	100%
8	<input checked="" type="checkbox"/> Documentación	Adrián Valdivieso...	15h	28/Apr	04/May	Finished	100%
9	<input checked="" type="checkbox"/> Instalar bootstrap, gestionar tema de aplicacion CLIENTE	Adrián Valdivieso...	15h	05/May	11/May	Finished	100%
10	<input checked="" type="checkbox"/> Inicio de sesion, Google Auth CLIENTE	Adrián Valdivieso...	30h	12/May	18/May	Finished	100%
11	<input checked="" type="checkbox"/> Interfaces de usuario CLIENTE	Adrián Valdivieso...	30h	19/May	25/May	Finished	100%
12	<input checked="" type="checkbox"/> Implementacion RDS AWS, Google Calendar	Adrián Valdivieso...	30h	26/May	01/Jun	Finished	50%
13	<input checked="" type="checkbox"/> Documentación	Adrián Valdivieso...	30h	02/Jun	08/Jun	Finished	100%

[+ Add task](#) [+ Add section](#)
*Diagrama 6 Desarrollo Gantt**Diagrama 7 Gantt*

## 10. Anexo 2 Pruebas de Funcionamiento.

10.1. La página tiene que tener un uso destinado para un usuario y un uso destinado para un profesional.

Para llevar a cabo esta prueba de funcionamiento se tiene que explicar previamente que se ha llegado a la conclusión de que era conveniente agregar un nuevo rol, de esta manera no se deja en manos de varias personas el mantenimiento de la misma.



The screenshot shows the 'Lista de Usuarios' (User List) page. At the top, there is a green header bar with the 'Plena Inclusión Aragón' logo and navigation buttons for 'Inicio', 'Ver Actividades', 'Mis Actividades', and 'Ver Usuarios'. Below the header, the title 'Lista de Usuarios' is displayed. There are four user cards: 1. Monitor: Oana Diana (oana@gmail.com). 2. Nominal: Adrian Valdivieso Paredes (adriavaldicorona@gmail.com). 3. Administrador: Adrián Valdivieso Paredes (adrianmvparedes@gmail.com). 4. Nominal: Adrian Valdi (kyudragonx@gmail.com). Each card has a 'Ver Perfil' button at the bottom. A sidebar message says: 'En esta página puedes ver a todos los usuarios registrados en la aplicación.'

### Pruebas de Funcionamiento 1

En esta página se pueden ver los distintos roles de los usuarios registrados en la página.

10.2. Registro de usuarios.



The screenshot shows the 'Nuevo Registro' (New Registration) page. It features a large input field for a profile picture with the placeholder 'Por favor seleccione una foto.'. Below it are fields for 'DNI' (obligatory), 'Fecha de Nacimiento' (obligatory), 'Rol' (obligatory), 'Nombre' (obligatory), 'Primer Apellido' (obligatory), 'Segundo Apellido' (obligatory), and 'Correo Electrónico' (obligatory). A sidebar message says: 'En esta página puedes registrar nuevos usuarios en la aplicación, asegurate de llenar todos los campos.'

### Pruebas de Funcionamiento 2

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | HTTP://ZARAGOZA.SALESIANOS.EDU





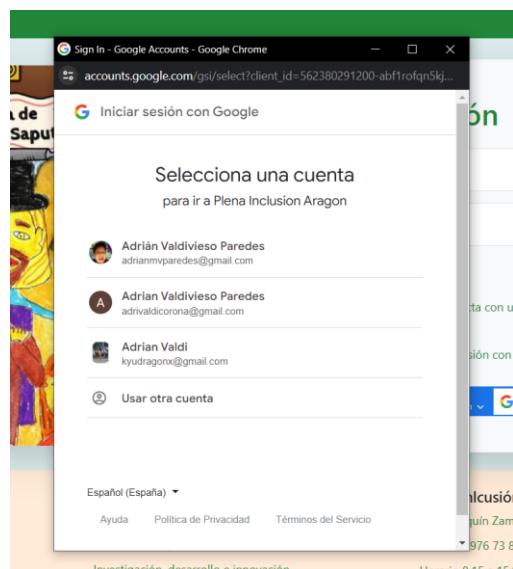
En esta página se presenta un formulario, el cual el usuario tendrá que llenar para poder ingresar un nuevo usuario a la aplicación.

### 10.3. Inicio de sesión.

A screenshot of a web browser showing the "Inicia Sesión" (Login) page of the Plena Inclusión Aragón website. The page has a green header with the "Plena Inclusión Aragón" logo. The main content area features a colorful illustration of a person with a speech bubble that says "La vida de Pedro Saputo". To the right is a login form with fields for "Correo Electrónico" and "Contraseña", a "Iniciar Sesión" button, and links for creating an account and Google sign-in. At the bottom, there's a sidebar with links for "Guía sobre discapacidad intelectual", "Accesibilidad cognitiva", and "Plena inclusión Aragón" information.

### Pruebas de Funcionamiento 3

El inicio de sesión en la aplicación se realiza mediante dos formas, la primera es mediante un registro previo y la segunda es mediante la autenticación de Google.

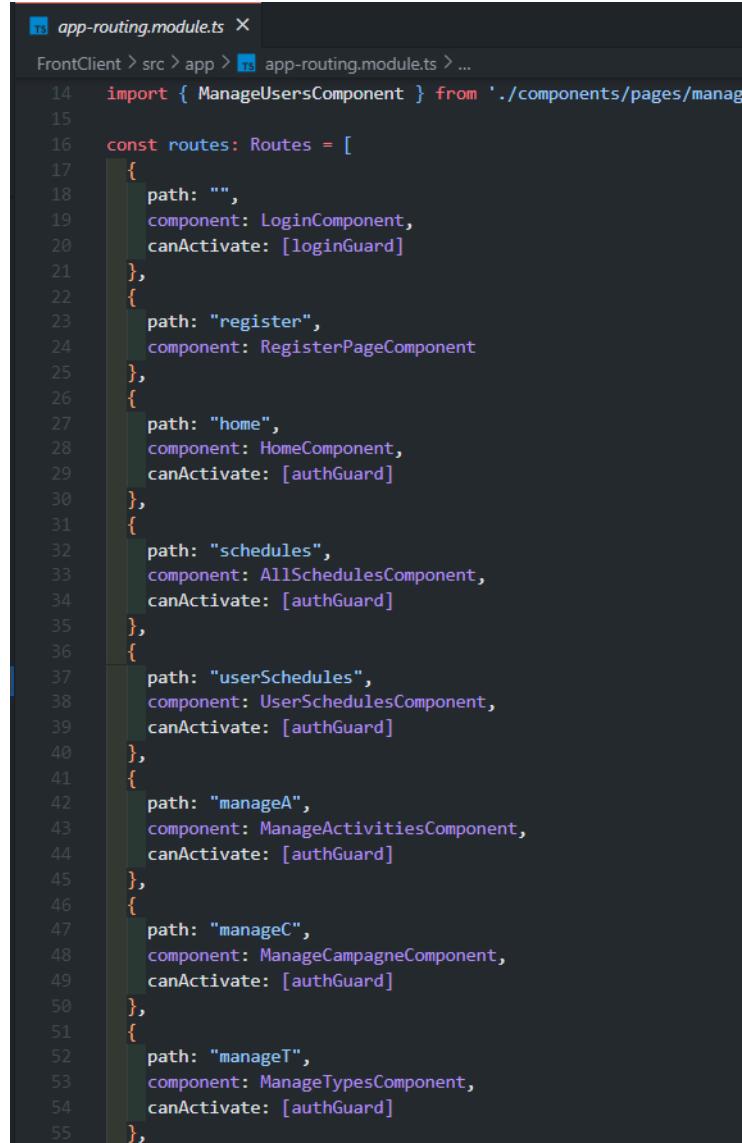


### Pruebas de Funcionamiento 4

Al pulsar el botón de inicio de sesión mediante Google, se abrirá una ventana nueva donde se podrá elegir la cuenta con la que se desea acceder.

10.4. Para acceder a todas las funcionalidades de la página se necesitará un registro previo.

Para comprobar esta funcionalidad se tiene que mostrar la configuración de las rutas dentro del proyecto:



```
app-routing.module.ts
FrontClient > src > app > app-routing.module.ts > ...
14 import { ManageUsersComponent } from './components/pages/manage...
15
16 const routes: Routes = [
17 {
18   path: '',
19   component: LoginComponent,
20   canActivate: [loginGuard]
21 },
22 {
23   path: "register",
24   component: RegisterPageComponent
25 },
26 {
27   path: "home",
28   component: HomeComponent,
29   canActivate: [authGuard]
30 },
31 {
32   path: "schedules",
33   component: AllSchedulesComponent,
34   canActivate: [authGuard]
35 },
36 {
37   path: "userSchedules",
38   component: UserSchedulesComponent,
39   canActivate: [authGuard]
40 },
41 {
42   path: "manageA",
43   component: ManageActivitiesComponent,
44   canActivate: [authGuard]
45 },
46 {
47   path: "manageC",
48   component: ManageCampaigneComponent,
49   canActivate: [authGuard]
50 },
51 {
52   path: "manageT",
53   component: ManageTypesComponent,
54   canActivate: [authGuard]
55 },
```

### Pruebas de Funcionamiento 5

Angular ofrece una funcionalidad (los *guards*) con la cual podemos proteger rutas según los parámetros que necesitamos, en este caso, si el usuario no está registrado, todas las páginas le redireccionarán a la página de inicio de sesión. A su vez, un usuario que ya tiene una sesión iniciada no podrá acceder a la página de inicio de sesión nuevamente.



## 10.5. Crear actividades para los usuarios.

Crear Horario de Actividad

Selecciona la Actividad

Patinaje Clases de patinaje para principiantes	Baloncesto mixto Partido de baloncesto con los compañeros y compañeras	Senderismo Paseo por la montaña en Monte Perdido	Lectura Grupo del lectura diversa

Seleccionar Campaña

Seleccionar Tipo

Frecuencia

Dirección

Fecha de Inicio dd/mm/aaaa

Fecha de Finalización dd/mm/aaaa

La fecha de inicio debe ser anterior a la fecha de finalización.

Hora de Inicio

Hora de Finalización

Cerrar Rellenar los Campos restantes

### Pruebas de Funcionamiento 6

Como se puede ver en la imagen, tanto el rol de monitor como el rol de administrador, podrán crear horarios en los cuales los usuarios se podrán apuntar. Rellenando correctamente los respectivos campos.

## 10.6. Todo tiene que estar en modo de lectura fácil.

Plena Inclusión Aragón

Inicio Ver Actividades Mis Actividades Ver Usuarios

¡¡¡Bienvenido!!!

Adrián Valdivieso Paredes adrianmvparedes@gmail.com

Ver Perfil Cerrar Sesión

Plena Inclusión Aragón ¿Qué es?

Inclusión Aragón es una iniciativa que promueve la integración y la igualdad de oportunidades para todas las personas en Aragón. Se centra en eliminar barreras y fomentar la participación activa en diversos ámbitos sociales, económicos y culturales.

Más sobre esta aplicación

Soy Participante ¿Qué puedo hacer?

Soy Monitor ¿Qué puedo hacer?

Soy Administrador ¿Qué puedo hacer?

Plena Inclusión Aragón Justo ahora

Recuerda cerrar sesión antes de irte.

Aquí puedes informarte sobre como usar la aplicación.

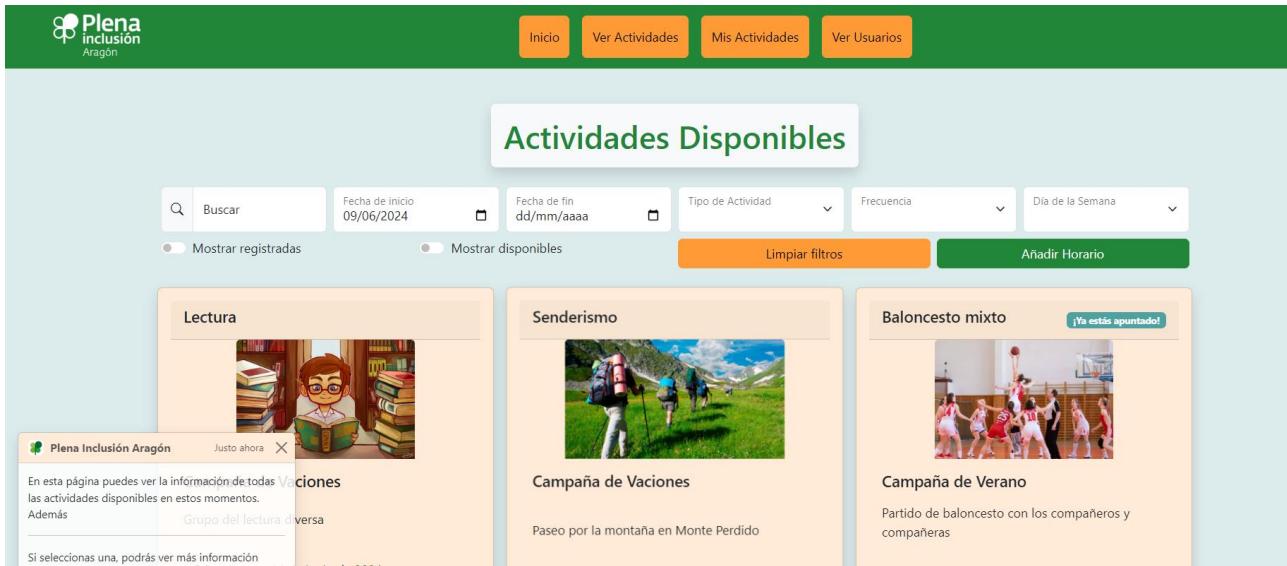
### Pruebas de Funcionamiento 7

Como se puede ver en la imagen, se ha utilizado el modo de lectura fácil, además se ha dotado a las páginas de mensajes informativos en la esquina inferior izquierda de cada página.

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)



### 10.7. Notificar a los usuarios de las actividades.



#### Pruebas de Funcionamiento 8

Como se puede ver en la imagen, los usuarios podrán ver una lista con todos los horarios disponibles, además estos podrán filtrar los resultados haciendo uso de las herramientas de filtrado presentes.

### 10.8. Lista de espera.



#### Pruebas de Funcionamiento 9

Ya que la esencia de esta organización se centra en la inclusión de las personas, se ha decidido modificar este requerimiento, de manera que, una actividad tendrá un máximo orientativo de asistentes, además se mostrara para cada horario la cantidad de personas que están apuntadas.



### 10.9. Crear formularios para la valoración de las actividades.

Comentarios de la Actividad: Baloncesto mixto

Comentarios en total: 2      Nota media de la actividad: 9

**Adrian Valdivieso Paredes**  
10 de junio de 2024      8/10  
"Estoy entusiasmado por esta actividad pero nervioso también porque no la he practicado nunca."

**Adrián Valdivieso Paredes**  
10 de junio de 2024      10/10  
"Me parece que esta actividad va a ayudar a muchos a salir de su zona de confort, además de poder pasar un buen rato."

**Cerrar**

### Pruebas de Funcionamiento 10

Como se puede ver en la imagen, los horarios tienen una sección donde los usuarios podrán ver los comentarios de los demás usuarios.

Además, si un usuario está apuntado a un horario, podrá agregar comentarios cada vez que asista a una actividad.

Comenta la Actividad

Nota  
10

Comentario  
Me parece que esta actividad va a ayudar a muchos a salir de su zona de confort, además de poder pasar un buen rato.

**Cerrar**      **Guardar cambios**

### Pruebas de Funcionamiento 11

### 10.10. Distintos programas.



The screenshot shows a mobile application interface for 'Plena Inclusión Aragón'. At the top, there is a green header bar with the logo and name. Below it is a white header bar with four orange buttons labeled 'Inicio', 'Ver Actividades', 'Mis Actividades', and 'Ver Usuarios'. On the right side of the white bar is a three-line menu icon. The main content area has a light blue background and features a title 'Lista de Campañas' (List of Campaigns) in green. Below the title is a green button labeled 'Anadir Campaña' (Add Campaign). Underneath is a search bar with a magnifying glass icon and the placeholder 'Buscar'. A table lists three campaigns with columns for 'Nombre' (Name), 'Descripción' (Description), 'Fecha de Inicio' (Start Date), and 'Fecha de Fin' (End Date). The campaigns are: 'Campaña de Vacaciones' (Vacation Campaign), 'Temporada para desconectar'; 'Campaña de Invierno' (Winter Campaign), 'Empieza el invierno, actividades más movidas'; and 'Campaña de Verano' (Summer Campaign), 'Ven a jugar con tus compañeros'. At the bottom of the screen, there is a notification bar with a message from 'Plena Inclusión Aragón' sent 'Justo ahora' (Just now) stating: 'En esta página puedes gestionar las campañas dentro de la aplicación.' (You can manage campaigns on this page within the application.)

#### Pruebas de Funcionamiento 12

En esta página se puede ver un listado con las campañas/programas existentes, estos programas irán asignados a cada horario, y estos horarios podrán ser filtrados por la campaña a la que pertenecen.



The screenshot shows a detailed view of a campaign. The title is 'Lectura' (Reading). Below the title is a cartoon illustration of a boy with glasses reading a book. The text 'Campaña de Vacaciones' (Vacation Campaign) is displayed below the illustration. At the bottom of the screen, there is a footer with the text 'Centro docente diverso' (Diverse teaching center).

#### Pruebas de Funcionamiento 13

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





10.11. Los profesionales podrán ver una lista de todos los usuarios registrados en la página web.

### Lista de Usuarios

[Registrar Usuario](#)

Monitor	Nominal	Administrador	Nominal
 Oana Diana <a href="#">Oana@gmail.com</a> <a href="#">Ver Perfil</a>	 Adrian Valdivieso Paredes <a href="#">adrivaldicorona@gmail.com</a> <a href="#">Ver Perfil</a>	 Adrián Valdivieso Paredes <a href="#">adriannvparedes@gmail.com</a> <a href="#">Ver Perfil</a>	 Adrian Valdi <a href="#">kyudragonx@gmail.com</a> <a href="#">Ver Perfil</a>

**Plena Inclusión Aragón Justo ahora** En esta página puedes ver a todos los usuarios registrados en la aplicación.

#### Pruebas de Funcionamiento 14

Los profesionales tendrán acceso a una lista con todos los usuarios registrados, al pulsar en uno de ellos podrán ver información más a detalle, además podrán eliminar un usuario si lo desean.

#### Información del Usuario



 DNI 45587463W	 Fecha de Nacimiento 25 de septiembre de 1999	 Rol: Administrador
 Nombre Adrián	 Primer Apellido Valdivieso Paredes	 Segundo Apellido
 Correo Electrónico adriannvparedes@gmail.com		
 Dirección Calle Domingo Ram 28	 Teléfono 644299785	
 Tutor 73515017W		

[Eliminar Usuario](#) [Cerrar](#)

#### Pruebas de Funcionamiento 15

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





## 10.12. Diseño corporativo.

**Plena inclusión Aragón**

**Actividades Disponibles**

Buscar | Fecha de inicio 09/06/2024 | Fecha de fin dd/mm/aaaa | Tipo de Actividad | Frecuencia | Día de la Semana | Inicio | Ver Actividades | Mis Actividades | Ver Usuarios | Añadir Horario | Limpiar filtros

Mostrar registradas  Mostrar disponibles

**Lectura**  
  
**Senderismo**  
  
**Baloncesto mixto**  
  
**Campaña de Vaciones**  
  
**Campaña de Verano**  
  
**Realización:** Jueves, 20 de junio de 2024  
**Inicio:** Lunes, 10 de junio de 2024  
**Finalización:** Lunes, 8 de julio de 2024  
**Asistentes:** 0/20 | Ya estás apuntado!

**Plena Inclusión Aragón**  
Justo ahora | Grupo del lectura | Versa | Asistentes: 0/30

En esta página puedes ver la información de todas las actividades disponibles en estos momentos. Además Si seleccionas una, podrás ver más información sobre ellas, comentarios e incluso apuntarte!!! Finalización: Viernes, 14 de junio de 2024. Puedes buscar una actividad en concreto utilizando los filtros.

### Pruebas de Funcionamiento 16

**Plena inclusión Aragón**

**Actividades Disponibles**

Buscar | Fecha de inicio 09/06/2024 | Fecha de fin dd/mm/aaaa | Tipo de Actividad | Frecuencia | Día de la Semana | Inicio | Ver Actividades | Mis Actividades | Ver Usuarios | Añadir Horario | Limpiar filtros

Mostrar registradas  Mostrar disponibles

**Lectura**  
  
**Senderismo**  
  
**Baloncesto mixto**  
  
**Campaña de Vaciones**  
  
**Campaña de Verano**  
  
**Realización:** Jueves, 20 de junio de 2024  
**Inicio:** Lunes, 10 de junio de 2024  
**Finalización:** Lunes, 8 de julio de 2024  
**Asistentes:** 0/20 | Ya estás apuntado!

**Plena Inclusión Aragón**  
Justo ahora | Grupo del lectura | Versa | Asistentes: 0/30

En esta página puedes ver la información de todas las actividades disponibles en estos momentos. Además Si seleccionas una, podrás ver más información sobre ellas, comentarios e incluso apuntarte!!! Finalización: Viernes, 14 de junio de 2024. Puedes buscar una actividad en concreto utilizando los filtros.

### Pruebas de Funcionamiento 17

La aplicación se realizó respetando los colores corporativos de la asociación, además se le ha agregado la funcionalidad del cambio de tema a modo oscuro.

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





### 10.13. Vista para el usuario con las actividades los siguientes siete días.

The screenshot shows a dark-themed user interface for 'Tu Horario'. At the top, there are search and filter fields for 'Buscar', 'Fecha de inicio' (09/06/2024), 'Fecha de fin' (16/06/2024), 'Tipo de Actividad' (Todos los tipos), and 'Día de la Semana' (Todos los días). Below these are buttons for 'Limpiar filtros' (Clear filters) and 'De hoy a siete días' (From today to seven days). A large card displays a basketball activity: 'Baloncesto mixto' on June 10, 2024, at the 'Pabellón Santa Isabel' from 10:00 to 11:00. It's part of the 'Campaña de Verano' and has 3 participants. A note says: 'Estas son las actividades que tienes pendientes de aquí a siete días, para ver todas pulsa en el botón "Limpiar filtros".' A sidebar on the left provides instructions for leaving a comment and searching for activities.

#### Pruebas de Funcionamiento 18

Esta página, en modo oscuro, muestra las siguientes actividades que tiene por realizar el usuario los siguientes 7 días, también puede limpiar los filtros para que aparezcan todas las actividades a las que tiene que asistir.

This screenshot shows the same 'Tu Horario' interface but with multiple activities listed. There are five cards, each representing a basketball session: one for June 10, 17, and 24, and two for July 1 and 8. Each card follows the same structure: activity type ('Baloncesto mixto'), date, location ('Pabellón Santa Isabel'), time ('10:00 - 11:00'), campaign ('Campaña de Verano'), and participants ('3'). Below each card are buttons for 'Tipo de Actividad: Aire Libre' and 'Semanal'.

#### Pruebas de Funcionamiento 19

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





### 10.14. Google Calendar.

Lamentablemente no se ha podido cumplir con esta funcionalidad, debido a diversos problemas.

El problema más importante ha sido establecer conexión con el servicio de Google que nos permite realizar esta tarea, da un conflicto de CORS (redireccionamiento de páginas), con el cual no se ha podido lidiar a tiempo.

A modo de respuesta a esta funcionalidad, se ha diseñado que cada vez que un usuario se registra en un horario, se le manda un correo informativo sobre las fechas a las que debe asistir y demás información relevante.

A screenshot of an email inbox in Spanish. The subject line is "Recordatorio Actividades" and the recipient is "adrianmvparedes@gmail.com" (para mí). The email body contains the following text:

Nombre de la actividad: Baloncesto mixto  
Debes acudir a la dirección: Pabellón Santa Isabel  
A la hora: 10:00:00  
Los días: Lunes  
Fechas:

- 2024-06-10
- 2024-06-17
- 2024-06-24
- 2024-07-01
- 2024-07-08

At the bottom of the email are three buttons: "Responder" (Reply), "Reenviar" (Forward), and a smiley face icon.

### Pruebas de Funcionamiento 20

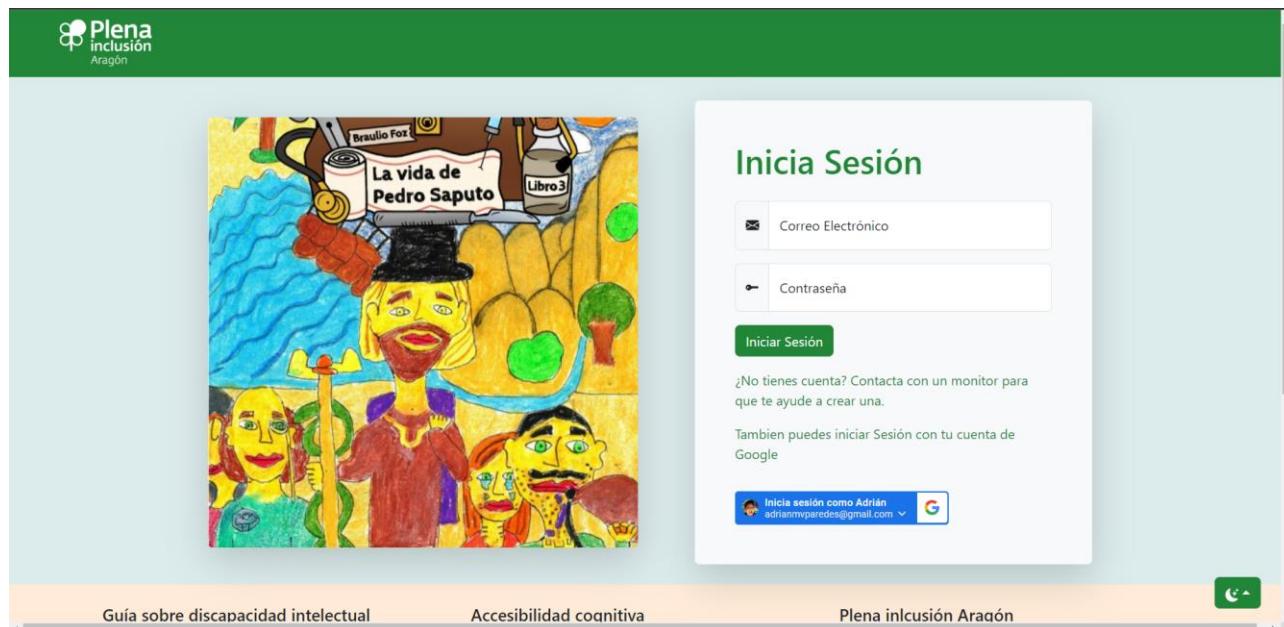
De esta manera la funcionalidad de notificar al usuario sobre las actividades en las que se apunta queda resuelta de alguna manera.

## 11. Anexo 3 Manual de Usuario.

### 11.1. Descripción del documento.

En este documento se enseña el funcionamiento básico de la aplicación a los distintos tipos de usuarios. Mostrando paso a paso cada una de las funcionalidades, como el registro, inicio de sesión, etc. Empezando por el rol *Nominal*, debido a que todos los roles comparten estas funcionalidades.

### 11.2. Inicio de sesión.



#### Manual de Usuario 1 Inicio de sesión

En esta pantalla se permite al usuario realizar el inicio de sesión, cuenta con dos maneras para llevarlo a cabo, la primera es introduciendo el usuario y contraseña (para lo cual tiene que haber un registro previo) y la segunda manera es mediante el uso de la autenticación de Google.

#### Inicia Sesión

#### Manual de Usuario 2 Registro fallido

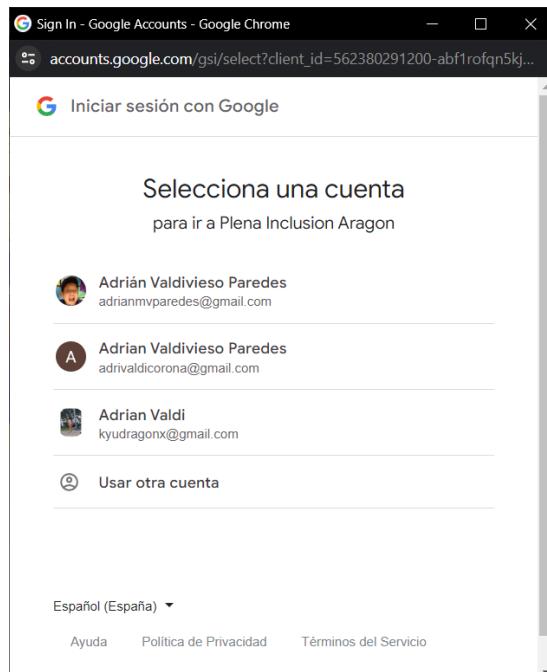
C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | HTTP://ZARAGOZA.SALESIANOS.EDU





Si el usuario introduce mal las credenciales de acceso, será informado sobre cual es el campo erróneo.

Para hacer el inicio de sesión mediante Google, se tiene que pulsar en el botón de Google, y el usuario será redirigido a la página de Google, donde tendrá que iniciar sesión. Así, de esa manera, el usuario tendrá acceso a la aplicación.



*Manual de Usuario 3 Inicio Sesión Con Google*



### 11.3. Página principal usuario Nominal.

A screenshot of a web application interface. At the top, there's a green header bar with the "Plena inclusión Aragón" logo, three orange buttons labeled "Inicio", "Ver Actividades", and "Mis Actividades", and a close button "X". The main content area has a light blue background. On the left, a sidebar titled "¡¡¡Bienvenido!!!" shows a green four-leaf clover icon, the user's name "Adrian Valdivieso Paredes", and their email "adivaldicorona@gmail.com". Below this are two buttons: "Ver Perfil" and "Cerrar Sesión". To the right of the sidebar is a section titled "Plena Inclusión Aragón ¿Qué es?" which contains text about the initiative's goals and a "Más sobre esta aplicación" button. Further down is a section titled "Soy Participante ¿Qué puedo hacer?" with a "Soy Participante" button. At the bottom of the sidebar, there's a message: "Recuerda cerrar sesión antes de irte." and "Aquí puedes informarte sobre como usar la aplicación." A small notification box at the bottom left says "Plena Inclusión Aragón Justo ahora X".

#### Manual de Usuario 4 Página Principal Nominal

Esta es la página principal que podrá ver el usuario que tenga el rol Nominal, en esta pagina se puede apreciar una pequeña sección con los datos del usuario registrados, junto con estos dos botones.

El primero servirá para revisar los datos del usuario y si es necesario editar los campos editables.

A modal dialog box titled "Editar Perfil". It features a large green four-leaf clover icon in the center. Below it are several input fields: "DNI" (73515017W), "Fecha de Nacimiento" (dd/mm/aaaa), and "Rol" (Nominal). Other fields include "Nombre" (Adrian), "Primer Apellido" (Valdivieso Paredes), "Segundo Apellido", "Correo Electrónico" (adivaldicorona@gmail.com), "Dirección" (Calle domingo ram), "Teléfono" (644299785), and a dropdown for "Seleccione un Tutor" (Adrian Valdi (569239565)). At the bottom are "Actualizar" and "Cerrar" buttons.

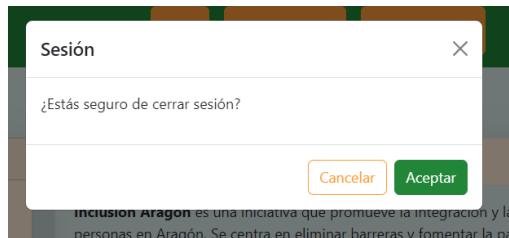
#### Manual de Usuario 5 Perfil del Usuario

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | HTTP://ZARAGOZA.SALESIANOS.EDU





El segundo botón servirá para cerrar la sesión, al pulsarlo se mostrará una ventana emergente la cual preguntará al usuario si está seguro de cerrar la sesión, si el usuario pulsa en cancelar, mantendrá la sesión, si pulsa Aceptar, cerrará la sesión.



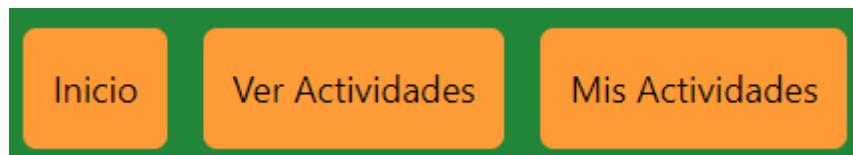
#### *Manual de Usuario 6 Confirmación de Cierre de Sesión*

En la sección de la derecha se sitúa un apartado a modo informativo, donde el usuario podrá informarse sobre la asociación y además podrá informarse sobre el uso de la aplicación.

A screenshot of a sidebar with three expandable sections: "Plena Inclusión Aragón ¿Qué es?", "Más sobre esta aplicación", and "Soy Participante ¿Qué puedo hacer?". The "Soy Participante" section is expanded, showing instructions for using the app. It includes a numbered list of steps: 1. Revisa todas las actividades disponibles en el apartado Ver Actividades. Puedes filtrar las actividades por nombre, tipo, fecha, etc. 2. Al seleccionar una actividad, podrás ver los comentarios de otros participantes y podrás apuntarte si te interesa. 3. En el apartado Mis Actividades, podrás ver las actividades a las que estás inscrito desde hoy hasta siete días adelante. 4. Deja una puntuación y un breve comentario sobre la actividad antes o después de asistir. Below this, a note states: "Esta aplicación te permitirá estar al tanto de todas las actividades que se desarrollan en nuestra asociación y te brindará la posibilidad de apuntarte a ellas. ¡No dudes en participar!"

#### *Manual de Usuario 7 Información General Nominal*

Además, en la barra de navegación se podrá distinguir tres botones, el primero sirve para regresar a esta página, el segundo botón lleva a los horarios disponibles y el tercer botón lleva a los horarios en los que el usuario está apuntado.



#### *Manual de Usuario 8 Opciones Nominal*



#### 11.4. Página principal usuario Monitor.

La página principal para el monitor comparte varios aspectos con la página principal para el usuario Nominal. El único aspecto que cambia es la información que se le otorga en el apartado de información sobre la aplicación.

A screenshot of a mobile application interface. At the top, there are four expandable sections: "Plena Inclusión Aragón ¿Qué es?", "Más sobre esta aplicación", "Soy Participante ¿Qué puedo hacer?", and "Soy Monitor ¿Qué puedo hacer?". The "Soy Monitor" section is expanded, revealing a sub-section titled "Ser Monitor en Inclusión Aragón implica más responsabilidad. Aquí te mostramos los pasos básicos:" followed by a numbered list of five steps: 1. Realiza las mismas acciones que un usuario normal. 2. Visualiza una lista con todos los usuarios registrados en la aplicación. 3. Al pulsar en cada carta de usuario, podrás ver más detalles sobre el usuario seleccionado. 4. Añade actividades al apartado de actividades. 5. Elimina actividades del calendario en el apartado Mis Actividades.

#### *Manual de Usuario 9 Información para el Monitor*

Además, en la barra de tareas, este usuario tendrá una opción adicional, el botón para ver todos los usuarios de la aplicación.



#### *Manual de Usuario 10 Opciones Monitor*



### 11.5. Página principal usuario Administrador.

La página principal para el administrador comparte varios aspectos con la página principal para el usuario Nominal y el usuario Monitor, se le agrega más información al apartado de información general mencionado antes.

The screenshot shows a mobile application interface. At the top, there is a navigation bar with five items: "Plena Inclusión Aragón ¿Qué es?", "Más sobre esta aplicación", "Soy Participante ¿Qué puedo hacer?", "Soy Monitor ¿Qué puedo hacer?", and "Soy Administrador ¿Qué puedo hacer?". The last item is currently selected and has an upward arrow icon next to it. Below this is a section titled "Como administrador, tienes el control total de la aplicación. Estas son tus ventajas:" followed by a numbered list of 5 points detailing the benefits of being an administrator. At the bottom of this section is the text "Tendrás la completa gestión de la aplicación."

*Manual de Usuario 11 Información Administrador*

Además, este usuario tendrá acceso al panel de control, donde podrá gestionar la aplicación más a fondo.

The screenshot shows the "Panel de Control" (Control Panel) of the application. At the top, there is a header with the "Plena inclusión Aragón" logo and the text "Panel de Control" and a close button ("X"). On the left, there is a sidebar with a button labeled "Ver Usuarios". The main area contains three buttons: "Gestión de Actividades", "Gestión de Campañas", and "Gestión de Tipos de Actividades".

*Manual de Usuario 12 Panel de Control*

## 11.6. Página de horarios Nominal.



The screenshot shows the 'Actividades Disponibles' (Available Activities) section of the website. It features three main activity cards:

- Lectura:** Shows a person reading a book. Description: Grupo del lectura diversa. Details: Justo ahora, 09/06/2024 - 19/07/2024, Realización: Jueves, 20 de junio de 2024, Asistentes: 0/20.
- Senderismo:** Shows people hiking in a mountainous area. Description: Campaña de Vaciones. Details: Paseo por la montaña en Monte Perdido, Realización: Jueves, 20 de junio de 2024.
- Baloncesto mixto:** Shows people playing basketball. Description: Campaña de Verano. Details: Partido de baloncesto con los compañeros y compañeras, Inicio: Lunes, 10 de junio de 2024, Finalización: Lunes, 8 de julio de 2024, Asistentes: 3/50.

At the top, there are search filters for 'Buscar', 'Fecha de inicio', 'Fecha de fin', 'Tipo de Actividad', 'Frecuencia', and 'Día de la Semana'. Below the filters are buttons for 'Mostrar registradas' and 'Mostrar disponibles', with 'Limpiar filtros' at the bottom right.

### Manual de Usuario 13 Página Horarios Nominal

En esta página el usuario podrá ver todos los horarios, y con ello tendrá varias opciones.

Dispondrá de un panel para filtrar todas las actividades según: el nombre de la actividad, la fecha de inicio, la fecha de finalización, el tipo de actividad, la frecuencia, y el día de la semana en el que se desarrolla, incluso podrá filtrar por actividades en las que está apuntado y actividades en las que no está apuntado. Al pulsar en una actividad podrá ver lo siguiente:



### Manual de Usuario 14 Información Horario

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





En este panel el usuario podrá tener más información sobre el horario en cuestión, además dispondrá de dos botones.

El botón de “Ver Comentarios”, este botón permitirá al usuario ver los comentarios que han sido agregados a dicho horario, además de contener la cantidad de comentarios y la media de la nota de todos los comentarios añadidos.



#### *Manual de Usuario 15 Comentarios horario*

De esta manera el usuario podrá decidir si apuntarse o no a la actividad mediante el botón “¡Me Apunto!”. Al pulsar el botón se mostrará una ventana emergente donde el usuario va a tener que confirmar su participación en la actividad.



#### *Manual de Usuario 16 Confirmar registro en horario*

Si el usuario pulsa en confirmar, se habrá apuntado en la actividad y recibirá un correo informativo con las fechas en las que tiene que asistir, donde tiene que asistir y más información sobre la actividad.



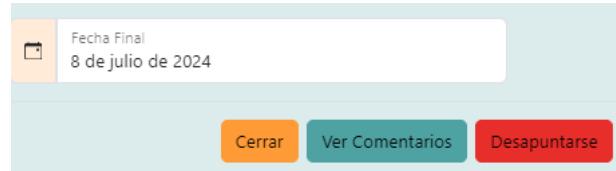
#### *Manual de Usuario 17 Correo Informativo*

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





Por último, el usuario tendrá la opción de desapuntarse de las actividades mediante el botón de “Desapuntarse”. Nuevamente esto desplegará una ventana para confirmar la solicitud del usuario.



*Manual de Usuario 18 Desapuntarse de Horario*

#### 11.7. Página de horarios Monitor y Administrador.

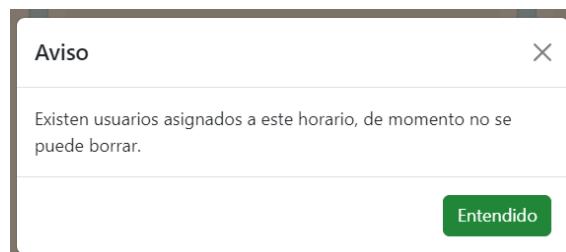
Además de compartir las funcionalidades que tiene el usuario Nominal, el usuario Monitor, y, a su vez el usuario Administrador, tendrá añadidas dos funcionalidades más, sobre gestión de estas actividades.

La primera capacidad es poder eliminar horarios mediante el botón, *Eliminar horario*.



*Manual de Usuario 19 Botón Eliminar horario*

Cabe destacar que sólo se podrá eliminar el horario si no hay usuarios apuntados a éste, si se intenta eliminar uno con estas condiciones se mostrará el siguiente mensaje:



*Manual de Usuario 20 Error Eliminar Horario*

La segunda es la capacidad de poder añadir nuevos horarios, mediante el botón de Añadir Horario.



*Manual de Usuario 21 Botón Añadir Horario*

Este botón desplegará una ventana donde el Monitor tendrá que llenar por completo para poder crear el horario en cuestión.

En el primer apartado se presentan un apartado donde se encuentran todas las actividades disponibles hasta el momento, tiene que seleccionar una.



Luego se presenta una sección con la información necesaria para crear el nuevo horario, en esta sección se podrá detallar: la campaña, el tipo de horario, la frecuencia, la dirección, si el horario es puntual solo se mostrará la fecha de inicio y si es semanal se mostrará la fecha de finalización. Por último, la hora de inicio y la hora de finalización, todos estos campos son necesarios.

Crear Horario de Actividad

Selección de Actividad

**Patinaje**  
Clases de patinaje para principiantes

**Baloncesto mixto**  
Partido de baloncesto con los compañeros y compañeras

**Senderismo**  
Paseo por la montaña en Monte Perdido

**Lectura**  
Grupo del lectura diversa

Seleccionar Campaña  
Campaña de Verano

Seleccionar Tipo

Frecuencia  
Puntual

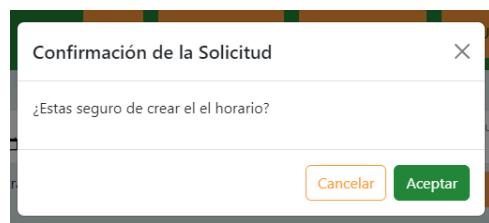
Hora de Inicio  
--:--

Hora de Finalización  
--:--

Capacidad  
0

### Manual de Usuario 22 Crear Horario

Una vez rellenados todos los campos, el usuario podrá presionar el botón de “Crear”, nuevamente aparecerá una ventana emergente para confirmar la solicitud, si el usuario confirma, el horario será creado, al contrario, si el usuario pulsa cancelar, no se creará el horario.



### Manual de Usuario 23 Confirmación Creación Horario



### 11.8. Pantalla Tu Horario.

En esta pantalla se muestran las actividades en las que el usuario se ha registrado, cabe destacar que esta pantalla se comparte entre los tres usuarios, no cambia de un usuario a otro

De entrada, se muestran sólo las actividades de la fecha actual a siete días por delante.

This screenshot shows the 'Tu Horario' (Your Schedule) page. At the top, there are search filters for 'Buscar', 'Fecha de inicio' (09/06/2024), 'Fecha de fin' (16/06/2024), 'Tipo de Actividad' (Todos los tipos), and 'Día de la Semana' (Todos los días). Below the filters are two buttons: 'Limpiar filtros' (Clear filters) and 'De hoy a siete días' (From today to seven days). The main content area displays a card for a basketball mixed activity on Monday, June 10, 2024. The card includes a thumbnail image of people playing basketball, the activity name 'Baloncesto mixto', the date 'Lunes 10 de junio de 2024', a description 'Partido de baloncesto con los compañeros y compañeras', location 'Ubicación: Pabellón Santa Isabel', time 'Horario: 10:00 -11:00', campaign 'Campaña: Campaña de Verano', and participants 'Participantes: 3'. At the bottom of the card, there are links for 'Tipo de Actividad: Aire Libre' and 'Semanal'.

#### *Manual de Usuario 24 Mis Horarios Nominal*

Pero mediante el botón de “Limpiar filtros”, se puede revisar todas las asistencias que tiene o ha tenido el usuario en la aplicación.

This screenshot shows the 'Tu Horario' (Your Schedule) page with the 'Limpiear filtros' (Clear filters) button selected, displaying all scheduled activities. It lists five basketball mixed activities across three dates: June 10, June 17, and June 24, 2024. Each activity card follows the same structure as the first one, providing details like date, description, location, time, campaign, and participants. At the bottom of each card, there are 'Tipo de Actividad: Aire Libre' and 'Semanal' links.

#### *Manual de Usuario 25 Horario Todas Asistencias*

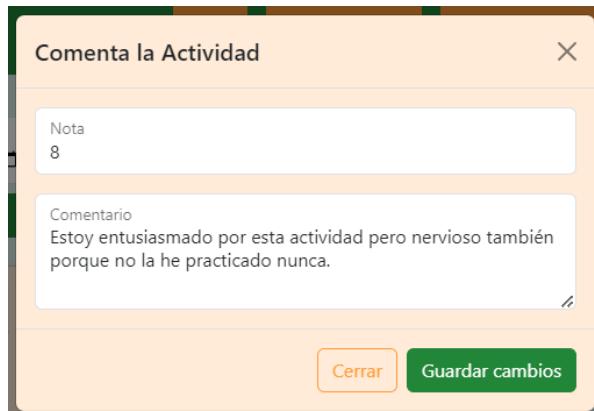
C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





El usuario tendrá las opciones de filtrado, donde podrá buscar cada asistencia por los siguientes aspectos: nombre, intervalo entre fecha de inicio y fecha final, tipo de actividad y día de la actividad.

Finalmente, el usuario podrá agregar una valoración y un breve comentario a la asistencia seleccionada pulsando sobre ella



#### *Manual de Usuario 26 Comentario Asistencia*

Esto quedará registrado en el apartado de comentarios en este horario, de esta manera los demás usuarios pueden informarse sobre la actividad.

Si el usuario se desapunta del horario, todas estas asistencias también quedarán eliminadas.



### 11.9. Pantalla ver usuarios Nominal y Administrador.

En esta pantalla se muestran todos los usuarios registrados en la aplicación

A screenshot of a web-based application interface. At the top, there's a green header bar with the "Plena Inclusión Aragón" logo on the left and four buttons: "Inicio", "Ver Actividades", "Mis Actividades", and "Ver Usuarios". On the right is a three-line menu icon. Below the header, the title "Lista de Usuarios" is centered in a green box. Underneath, there are three cards: "Nominal" (with a green leaf icon), "Administrador" (with a person icon), and another "Nominal" (with a person icon). Each card displays a user profile: "Adrian Valdivieso Paredes" (adrivaldicorona@gmail.com) in the first, "Adrián Valdivieso Paredes" (adrianmparedes@gmail.com) in the second, and "Adrian Valdi" (kyudragonx@gmail.com) in the third. Each profile has a "Ver Perfil" button below it. A small sidebar on the left says "Justo ahora" and "En esta página puedes ver a todos los usuarios registrados en la aplicación." A "c -" icon is in the bottom right corner.

### Manual de Usuario 27 Pantalla Usuarios

Al pulsar sobre cualquier perfil, se desplegará la información del usuario en cuestión.

A modal window titled "Información del Usuario". It features a large central portrait photo of a man. Below the photo are several data fields in light blue boxes: "DNI" (45587463W), "Fecha de Nacimiento" (25 de septiembre de 1999), "Rol" (Administrador); "Nombre" (Adrián), "Primer Apellido" (Valdivieso Paredes), "Segundo Apellido"; "Correo Electrónico" (adrianmparedes@gmail.com); "Dirección" (Calle Domingo Ram 28); "Teléfono" (644299785); and "Tutor" (73515017W). At the bottom right of the modal are two buttons: "Eliminar Usuario" (in red) and "Cerrar" (in green).

### Manual de Usuario 28 Información Usuario

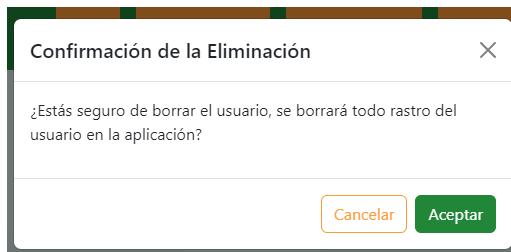
C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | HTTP://ZARAGOZA.SALESIANOS.EDU





En esta ventana se mostrarán los datos del usuario, además, se presentará el botón para eliminar al usuario, hay que tener mucho cuidado porque esto borrará cualquier rastro en la aplicación, incluidos sus comentarios.

Al pulsar en el botón “Eliminar Usuario”, se mostrará una ventana emergente para confirmar la solicitud, si el usuario acepta, borrará el registro del usuario seleccionado, sino lo mantendrá.



#### *Manual de Usuario 29 Confirmación Usuario Borrado*

#### 11.10. Pantalla lista de actividades Administrador.

En esta pantalla se le permite al administrador tener un control de las actividades que están vigentes en la aplicación. A esta pantalla se accede mediante el Panel de Control situado en la parte superior derecha.

A screenshot of a web application interface. The top navigation bar is green with the "Plena Inclusión Aragón" logo and links for "Inicio", "Ver Actividades", "Mis Actividades", and "Ver Usuarios". The main content area has a title "Lista de Actividades". Below it is a table with columns "Foto", "Nombre", and "Descripción". Four activities are listed: Patinaje (Clases de patinaje para principiantes), Baloncesto mixto (Partido de baloncesto con los compañeros y compañeras), Senderismo (Paseo por la montaña en Monte Perdido), and Lectura (Grupo del lectura diversa). A search bar labeled "Buscar" is above the table. A notification at the bottom left says "Plena Inclusión Aragón Justo ahora" and "En esta página puedes gestionar las actividades dentro de la aplicación." A small footer icon is at the bottom right.

#### *Manual de Usuario 30 Pantalla Lista de Actividades*

El usuario tendrá la posibilidad pulsar en cualquier actividad, esto le mostrará información a detalle de la misma, además será capaz de modificar dicha información, su foto, su título y su descripción.

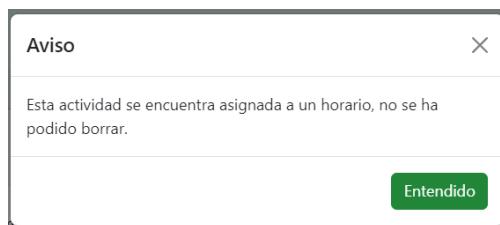
También podrá buscar actividades por nombre en la sección de búsqueda.



Editar Actividad

#### *Manual de Usuario 31 Modificar Actividad*

Además, dentro de esta pantalla, el usuario podrá eliminar la actividad en cuestión, cabe señalar que no se podrán eliminar actividades que ya estén asignadas a un horario. Si se intenta eliminar una actividad de estas características el usuario será informado debidamente.



#### *Manual de Usuario 32 Error Eliminación Actividad*

De igual manera el usuario será capaz de agregar nuevas actividades mediante el botón de “Añadir Actividad”, situado en la parte superior de la tabla. El usuario deberá llenar correctamente todos los apartados para la creación de la actividad.

Crear Actividad

#### *Manual de Usuario 33 Creación Actividad*



### 11.11. Pantalla lista de campañas Administrador.

En esta pantalla se le permite al administrador tener un control de todas las campañas que se encuentran vigentes en la aplicación, muestra una lista con todas las campañas disponibles, además muestra un buscador por si no se encuentra a primera vista la campaña deseada.

The screenshot shows a green header bar with the "Plena inclusión Aragón" logo and navigation buttons for "Inicio", "Ver Actividades", "Mis Actividades", and "Ver Usuarios". Below the header is a white box titled "Lista de Campañas" (Campaign List). Inside this box is a green button labeled "Añadir Campaña" (Add Campaign) and a search bar with a magnifying glass icon and the placeholder "Buscar". A table lists two campaigns:

Nombre	Descripción	Fecha de Inicio	Fecha de Fin
Campaña de Vacaciones	Temporada para desconectar	9 de junio de 2024	25 de agosto de 2024
Campaña de Verano	Ven a jugar con tus compañeros	8 de junio de 2024	6 de septiembre de 2024

At the bottom of the main content area, there is a small notification box: "Plena Inclusión Aragón Justo ahora X" with the message "En esta página puedes gestionar las campañas dentro de la aplicación." To the right of the notification is a green "C" icon with a downward arrow.

### Manual de Usuario 34 Pantalla Campañas

El usuario también podrá pulsar sobre una campaña y podrá editar información de la misma, además en la misma ventana se le da la opción de borrar la campaña.

The modal window is titled "Editar Campaña". It contains four input fields with icons: "Nombre" (Name) with the value "Campaña de Vacaciones", "Descripción" (Description) with the value "Temporada para desconectar", "Fecha de Inicio" (Start Date) with the value "09/06/2024", and "Fecha de Fin" (End Date) with the value "25/08/2024". At the bottom of the modal are three buttons: "Borrar Campaña" (Delete Campaign) in red, "Actualizar" (Update) in green, and "Cerrar" (Close) in red.

### Manual de Usuario 35 Editar Campaña



De igual manera, si el usuario intenta eliminar una campaña que tiene horarios asignados, recibirá un mensaje de error, al contrario, podrá eliminar la campaña sin ningún problema.



#### *Manual de Usuario 36 Error Eliminación Campaña*

En el botón de “Añadir Campaña” se le permite al usuario añadir una nueva campaña, cuando el usuario pulse sobre este botón, se le abrirá una nueva ventana, entonces deberá llenar correctamente los datos y seguidamente pulsar el botón de crear. De esta manera la campaña se creará correctamente.



#### *Manual de Usuario 37 Crear Campaña*



#### *Manual de Usuario 38 Confirmación Creación Campaña*



### 11.12. Pantalla lista de tipos de actividades Administrador.

En esta pantalla se le permite al usuario tener una lista de todos los tipos de actividades vigentes dentro de la aplicación.

A screenshot of a web application interface. At the top, there's a green header bar with the "Plena inclusión Aragón" logo on the left and four buttons: "Inicio" (Home), "Ver Actividades" (View Activities), "Mis Actividades" (My Activities), and "Ver Usuarios" (View Users). On the far right of the header is a three-line menu icon. The main content area has a title "Lista de Tipos de Actividades" in a green box. Below it is a button labeled "Añadir Tipo" (Add Type) and a search bar with a magnifying glass icon and the placeholder "Buscar". A table follows, with columns "Nombre" (Name) and "Descripción" (Description). The data rows are: "Aire Libre" (This activity develops outdoors), "Cubierto" (This activity is carried out in a hall), and "Mixto" (Activity that is carried out both outdoors and indoors). At the bottom of the page, a small notification box says "Plena Inclusión Aragón Justo ahora" and "En esta página puedes gestionar los tipos a los que pertenecen las actividades." There's also a "Cerrar" (Close) button in the bottom right corner of the notification.

#### Manual de Usuario 39 Pantalla Tipos de Actividades

En la parte superior se presentan dos botones, el primero se utiliza para agregar un nuevo registro de tipo de actividad. Y el segundo apartado permitirá al usuario buscar por el nombre del tipo.

El usuario deberá llenar correctamente todos los campos y seguidamente podrá crear la actividad.

A modal window titled "Crear Tipo de Actividad" (Create Activity Type). It contains two input fields: "Nombre del Tipo" (Type Name) with a placeholder "ABC" and "Descripción" (Description) with a placeholder "e e e e". At the bottom are two buttons: "Cerrar" (Close) on the left and "Rellena los Campos" (Fill Fields) on the right.

#### Manual de Usuario 40 Crear Tipo de Actividad

C/ MARÍA AUXILIADORA 57 | 50009 ZARAGOZA  
T 976 306 878 | [HTTP://ZARAGOZA.SALESIANOS.EDU](http://ZARAGOZA.SALESIANOS.EDU)





De igual manera, en la lista de tipos de actividades, el usuario podrá pulsar sobre una, esto le arbirá una ventana donde tendrá dos opciones, la primera es ser capaz de modificar el tipo de actividad.

Para ello deberá modificar los campos correspondientes de manera correcta y deberá pulsar el botón de “Actualizar”.

A screenshot of a modal dialog box titled "Editar Tipo". It contains two input fields: "Nombre" (Name) with the value "Aire Libre" and "Descripción" (Description) with the value "Esta actividad se desarrolla al aire libre". At the bottom are three buttons: "Borrar Tipo" (Delete Type) in red, "Actualizar" (Update) in green, and "Cerrar" (Close) in orange.

#### *Manual de Usuario 41 Editar Tipo de Actividad*

Seguidamente se le mostrará una ventana donde el usuario tendrá que confirmar su acción.



#### *Manual de Usuario 42 Confirmación Modificar Tipo de Actividad*

Si el usuario acepta, el registro será modificado, en caso contrario, el registro no se verá afectado.



## 12. Anexo 4 Documentación de la API.

Se ha generado mediante *POSTMAN* toda la documentación de la API, se puede visitar y revisar este documento en la siguiente dirección URL:

<https://documenter.getpostman.com/view/32205135/2sA3XLFjEz>

Esta documentación contiene ejemplos de como realizar solicitudes a los diferentes *End-Points* de la aplicación.

## 13. Anexo 5 Código fuente.

Debido a que se ha generado bastantes archivos, se ha decidido adjuntar el repositorio donde se ha subido el proyecto, este repositorio se encuentra en Git Hub y está abierto para que cualquier persona lo pueda revisar, se accede a él mediante el siguiente enlace:

<https://github.com/Adriansotte/PlenaInclusion>

## 14. Índice de imágenes

Imagen 1 Angular.....	8
Imagen 2 NodeJS .....	8
Imagen 3 Sequelize .....	8
Imagen 4 MySQL.....	9
Imagen 5 Draw.io .....	9
Imagen 6 GitHub.....	9
Imagen 7 Visual Studio Code.....	10
Imagen 8 PostMan.....	10
Imagen 9 Amazon Web Service .....	10
Imagen 10 Bootstrap.....	11
Imagen 11 Instagantt I.....	11
Imagen 12 Angular y React.....	12
Imagen 13 API REST.....	13
Imagen 14 Usuario Nominal.....	17
Imagen 15 Usuario Monitor.....	17
Imagen 16 Usuario Administrador .....	18
Imagen 17 Ejemplo GUID .....	20
Imagen 18 Estructura de Carpetas en el Servidor.....	27
Imagen 19 app.js .....	28
Imagen 20 Variables de Entorno .....	29
Imagen 21 Configuración Base de Datos .....	30
Imagen 22 Carpeta Models .....	30
Imagen 23 Modelo Usuario .....	31
Imagen 24 Soft Delete .....	31
Imagen 25 Archivo de Relaciones .....	32
Imagen 26 Carpeta Routes .....	32
Imagen 27 End Points de Usuario.....	33
Imagen 28 Carpeta Middlewares .....	34
Imagen 29 Token Middleware .....	34
Imagen 30 Middeware de Verificación de Rol .....	35
Imagen 31 Carpeta Controllers .....	35

Imagen 32 Controller GET .....	36
Imagen 33 Controller GetByID .....	36
Imagen 34 Controller POST .....	37
Imagen 35 Controller PUT .....	37
Imagen 36 Controller DELETE.....	38
Imagen 37 Carpeta Utils .....	38
Imagen 38 Error Útil.....	39
Imagen 39 Token Útil .....	39
Imagen 40 Encrypt Útil.....	40
Imagen 41 Storage Util .....	41
Imagen 42 Validator Util.....	41
Imagen 43 Carpeta Validators.....	42
Imagen 44 Validador User.....	42
Imagen 45 Estructura Angular.....	43
Imagen 46 index Plena Iclusion .....	44
Imagen 47 Paleta de colores .....	44
Imagen 48 Carpeta Components .....	45
Imagen 49 Componente de registro.....	45
Imagen 50 Carpeta Guards.....	45
Imagen 51 Carpeta Interceptors.....	46
Imagen 52 Servicio en Angular .....	46
Imagen 53 Modelo Campaign .....	47
Imagen 54 Carpeta Pipes .....	47
Imagen 55 Archivo Environments.....	47
Imagen 56 Carpeta Utils .....	48
Imagen 57 Servicio RDS.....	48
Imagen 58 Servicio RDS BBDD .....	49
Imagen 59 Motor de base de datos.....	49
Imagen 60 Plantilla de base de datos .....	50
Imagen 61 Habilitar Acceso Publico .....	50
Imagen 62 Costes RDS .....	50
Imagen 63 Servicio RDS Creado .....	51
Imagen 64 Error de Conexión .....	51

Imagen 65 Reglas de Entrada.....	51
Imagen 66 RDS en Funcionamiento .....	52
Imagen 67 Slack .....	54
Imagen 68 Tailwindcss .....	55
Imagen 69 ShadCN .....	55
Imagen 70 Git Graph .....	56

## 15. Índice de diagramas

Diagrama 1 Entidad Relación Previo.....	60
Diagrama 2 Entidad Relación Actual .....	61
Diagrama 3 Relacionable Previo.....	62
Diagrama 4 Relacionable Actual .....	63
Diagrama 5 Caso de Usos .....	64
Diagrama 6 Desarrollo Gantt .....	65
Diagrama 7 Gantt.....	65

## 16. Índice de pantallas.

Manual de Usuario 1 Inicio de sesión.....	77
Manual de Usuario 2 Registro fallido.....	77
Manual de Usuario 3 Inicio Sesión Con Google .....	78
Manual de Usuario 4 Página Principal Nominal .....	79
Manual de Usuario 5 Perfil del Usuario .....	79
Manual de Usuario 6 Confirmación de Cierre de Sesión .....	80
Manual de Usuario 7 Información General Nominal .....	80
Manual de Usuario 8 Opciones Nominal .....	80
Manual de Usuario 9 Información para el Monitor .....	81
Manual de Usuario 10 Opciones Monitor .....	81
Manual de Usuario 11 Información Administrador .....	82
Manual de Usuario 12 Panel de Control.....	82
Manual de Usuario 13 Página Horarios Nominal.....	83
Manual de Usuario 14 Información Horario .....	83
Manual de Usuario 15 Comentarios horario .....	84
Manual de Usuario 16 Confirmar registro en horario .....	84
Manual de Usuario 17 Correo Informativo.....	84

Manual de Usuario 18 Desapuntarse de Horario .....	85
Manual de Usuario 19 Botón Eliminar horario .....	85
Manual de Usuario 20 Error Eliminar Horario.....	85
Manual de Usuario 21 Botón Añadir Horario .....	85
Manual de Usuario 22 Crear Horario.....	86
Manual de Usuario 23 Confirmación Creación Horario .....	86
Manual de Usuario 24 Mis Horarios Nominal .....	87
Manual de Usuario 25 Horario Todas Asistencias .....	87
Manual de Usuario 26 Comentario Asistencia.....	88
Manual de Usuario 27 Pantalla Usuarios .....	89
Manual de Usuario 28 Información Usuario .....	89
Manual de Usuario 29 Confirmación Usuario Borrado .....	90
Manual de Usuario 30 Pantalla Lista de Actividades .....	90
Manual de Usuario 31 Modificar Actividad .....	91
Manual de Usuario 32 Error Eliminación Actividad .....	91
Manual de Usuario 33 Creación Actividad.....	91
Manual de Usuario 34 Pantalla Campañas .....	92
Manual de Usuario 35 Editar Campaña .....	92
Manual de Usuario 36 Error Eliminación Campaña.....	93
Manual de Usuario 37 Crear Campaña.....	93
Manual de Usuario 38 Confirmación Creación Campaña.....	93
Manual de Usuario 39 Pantalla Tipos de Actividades .....	94
Manual de Usuario 40 Crear Tipo de Actividad .....	94
Manual de Usuario 41 Editar Tipo de Actividad .....	95
Manual de Usuario 42 Confirmación Modificar Tipo de Actividad .....	95

## 17. Índice de pruebas de funcionamiento.

Pruebas de Funcionamiento 1 .....	66
Pruebas de Funcionamiento 2 .....	66
Pruebas de Funcionamiento 3 .....	67
Pruebas de Funcionamiento 4 .....	67
Pruebas de Funcionamiento 5 .....	68
Pruebas de Funcionamiento 6 .....	69



Pruebas de Funcionamiento 7 .....	69
Pruebas de Funcionamiento 8 .....	70
Pruebas de Funcionamiento 9 .....	70
Pruebas de Funcionamiento 10 .....	71
Pruebas de Funcionamiento 11.....	71
Pruebas de Funcionamiento 12 .....	72
Pruebas de Funcionamiento 13 .....	72
Pruebas de Funcionamiento 14 .....	73
Pruebas de Funcionamiento 15 .....	73
Pruebas de Funcionamiento 16 .....	74
Pruebas de Funcionamiento 17 .....	74
Pruebas de Funcionamiento 18 .....	75
Pruebas de Funcionamiento 19 .....	75
Pruebas de Funcionamiento 20 .....	76